

InPlant SCADA

脚本编辑器 使用手册






声 明

- 严禁转载本手册的部分或全部内容。
- 在不经预告和联系的情况下，本手册的内容有可能发生变更，请谅解。
- 本手册所记载的内容，不排除有误记或遗漏的可能性。如对本手册内容有疑问，请与我公司联系，联系邮箱：SMS@supcon.com。

商 标

中控、SUPCON、PLANTMATE、AI-POET、InPlant、dOps、ESP-iSYS、Webfield、ics、MultiF、SupField、APC 等均是中控技术股份有限公司注册商标，拥有商标的所有权。未经中控技术股份有限公司的书面授权，任何个人及企业不得擅自使用上述商标。对于非法使用我司商标的行为，我司将保留依法追究行为人及企业的法律责任的权利。

文档标志符定义

	<p>警告：标示有可能导致人身伤亡或设备损坏的信息。</p> <p>WARNING: Indicates information that a potentially hazardous situation which, if not avoided, could result in serious injury or death.</p>
	<p>电击危险：标示有可能产生电击危险的信息。</p> <p>RISK OF ELECTRICAL SHOCK: Indicates information that Potential shock hazard where HAZARDOUS LIVE voltages greater than 30V RMS, 42.4V peak, or 60V DC may be accessible.</p>
	<p>防止静电：标示防止静电损坏设备的信息。</p> <p>ESD HAZARD: Indicates information that Danger of an electro-static discharge to which equipment may be sensitive. Observe precautions for handling electrostatic sensitive devices</p>
	<p>注意：提醒需要特别注意的信息。</p> <p>ATTENTION: Identifies information that requires special consideration.</p>
	<p>提示：标记对用户的建议或提示。</p> <p>TIP: Identifies advice or hints for the user.</p>

目 录

脚本编辑器	1
1 概述	1
1.1 脚本语言简介	1
1.2 功能特点	1
2 主界面	2
2.1 标题栏	2
2.2 菜单栏和工具栏一览	2
2.2.1 文件菜单	2
2.2.2 编辑菜单	3
2.2.3 调试菜单	3
2.2.4 设置菜单	4
2.2.5 视图菜单	4
2.2.6 帮助菜单	4
2.3 右键菜单一览	5
2.4 工程窗口	6
2.5 输出/变量/堆栈/表达式窗口	6
2.6 对象浏览器/函数库窗口	7
2.7 拖动和自动隐藏窗口	8
3 基本操作	10
3.1 创建脚本	10
3.2 查找内容	10
3.3 替换内容	10
3.4 剪切、复制和粘贴脚本	11
3.5 撤销更改	11
3.6 编辑环境设置	11
3.7 编程助手	12
3.8 位号选择器功能	14
3.9 添加函数和事件向导	15
3.10 事件向导工具栏	16
3.11 自定义函数	16
3.12 脚本调试说明	18
4 脚本引用类型	20
5 脚本调试信息输出函数	20
5.1 Enable	21
5.2 Trace	21
5.3 LogMsg	21
5.4 LogFile	22

6 窗体对象函数	23
6.1 AccessLevel	23
6.2 Alarm	24
6.3 ClosePopPic	24
6.4 GetTagOffColor	25
6.5 GetTagOffDesc	25
6.6 GetTagOnColor	26
6.7 GetTagOnDesc	27
6.8 IsDomainTag	28
6.9 Login	28
6.10 Logout	29
6.11 OpenPic	29
6.12 PrintScreen	30
6.13 Quality	30
6.14 ShowLayer	31
6.15 ShowWindow	31
6.16 ShutDown	32
6.17 Tag	32
6.18 TagDecimal	32
6.19 TagDescription	33
6.20 TagN	33
6.21 TagName	34
6.22 GoForward	34
6.23 GoBack	35
6.24 GotoFatherNodeHomePic	35
6.25 GotoChildNodeHomePic	36
6.26 GotoPrePic	36
6.27 GotoNextPic	37
6.28 GotoNodeHomePic	37
6.29 GotoPreNodeHomePic	37
6.30 GotoNextNodeHomePic	38
6.31 GotoPreNodeThisPic	38
6.32 GotoNextNodeThisPic	39
7 APP 对象函数	40
7.1 Alarm	40
7.2 Database	42
7.3 Exit	43
7.4 GetCurrentUser	44

7.5	GetCurrentUserGrade	45
7.6	GetGroup	45
7.7	GetServerState	46
7.8	GetTagLimitHigh	46
7.9	GetTagLimitLower	47
7.10	GetTagUnitByName	47
7.11	GetTickCount	48
7.12	GetUser	48
7.13	GetUserCount	49
7.14	GetUserTeam	49
7.15	GetUserTeamCount	50
7.16	IsPrimaryServer	51
7.17	Log	51
7.18	Login	52
7.19	Logout	52
7.20	MsgBox	53
7.21	Mute	54
7.22	OpenPic	55
7.23	PopupPic	55
7.24	PopupPicEx	56
7.25	PrintScreen	56
7.26	ReadTag	56
7.27	ReadTagEx	57
7.28	ReadTagAsy	58
7.29	ReadTags	58
7.30	ReadValue	59
7.31	SaveValue	60
7.32	ServerInfo	60
7.33	SetPrimaryServerIP	61
7.34	WriteLog	62
7.35	WriteTag	62
7.36	WriteTagEx	63
7.37	WriteTagL	63
8	图形对象函数	64
8.1	公共接口	64
8.1.1	Bottom	64
8.1.2	Height	64
8.1.3	Layer	65

8.1.4 Left	65
8.1.5 Right	65
8.1.6 Top	65
8.1.7 Visible	66
8.1.8 Width	66
8.2 常用接口	66
8.2.1 Angle	66
8.2.2 Align	67
8.2.3 BackgroundColor	67
8.2.4 BackgroundStyle	67
8.2.5 Caption	68
8.2.6 CenterX	68
8.2.7 CenterY	68
8.2.8 EdgeColor	68
8.2.9 EdgeStyle	69
8.2.10 EdgeWidth	69
8.2.11 Enabled	69
8.2.12 Font	70
8.2.13 FontColor	70
8.2.14 GradientColor	70
8.2.15 GradientStep	70
8.2.16 GradientStyle	71
8.2.17 IsAutoSize	71
8.2.18 IsTransparent	71
8.2.19 IsSemiTransparent	72
8.2.20 LineColor	72
8.2.21 LineStyle	72
8.2.22 LineWidth	72
8.2.23 Picture	73
8.2.24 RotateCenterX/RotateCenterY	73
8.2.25 SetTitle	73
8.2.26 SetTagValue	74
8.2.27 TagName	74
8.2.28 Text	74
8.2.29 TextColor	74
8.2.30 TransparentColor	75
8.3 “管道（Pipe）”专用接口	75
8.3.1 ColorInner	75

8.3.2 ColorOutter	75
8.3.3 PipeWidth	76
8.4 “面板（Panel）”专用接口	76
8.5 “按钮（Button）”专用接口	76
8.5.1 ButtonStyle	76
8.5.2 IsManagedButton	77
8.5.3 IsSwitchButton	77
8.5.4 PicturePosition	77
8.6 “图片（Image）”专用接口	78
8.7 “组合框（RadioBox）”专用接口	78
8.7.1 CurrentSelectedBtn	78
8.7.2 GetTagValue	79
8.8 “菜单（Menu）”专用接口	79
8.9 “定时器控件（Timer）”专用接口	79
8.9.1 Enable	79
8.9.2 Interval	80
8.9.3 举例	80
8.10 “实时报警控件（Alarm）”专用接口	80
8.10.1 AckCurList	80
8.10.2 AckOne	81
8.10.3 GetAlmCount	81
8.10.4 SetFilterTag	82
8.11 “趋势控件（TrdGraph）”专用接口	82
8.11.1 AddPen	82
8.11.2 AddRDBPen	83
8.11.3 InsertPen	84
8.11.4 InsertRDBPen	84
8.11.5 DeletePen	85
8.11.6 SetPen	85
8.11.7 SetRDBPen	86
8.11.8 GetPenCount	86
8.11.9 GetBackColor	87
8.11.10 SetBackColor	87
8.11.11 GetFrontColor	88
8.11.12 SetFrontColor	88
8.11.13 GetTextColor	89
8.11.14 SetTextColor	89
8.11.15 GetBarLineColor	89

8.11.16 SetBarLineColor	90
8.11.17 GetDisplayDateTime	90
8.11.18 SetDisplayDateTime	91
8.11.19 ShowStdLines	91
8.11.20 HideStdLines	92
8.11.21 SetRDBInfo	92
8.11.22 SetRDBTimeField	93
8.11.23 Refresh	93
8.11.24 SetStdLineColor	94
8.11.25 GetStdLineColor	94
8.11.26 GetStdLineShowMode	95
8.11.27 SetStdLineShowMode	95
8.11.28 SetTrdShowType	96
8.11.29 GetTrdShowType	96
8.11.30 GetPen	97
8.11.31 GetRDBPen	97
8.12 下拉框控件 (ComboBox) 专用接口	98
8.12.1 下拉框 ComboBox 的事件函数 Change	98
8.12.2 AddItem	98
8.12.3 AddString	99
8.12.4 Clear	100
8.12.5 FindString	100
8.12.6 GetCount	100
8.12.7 GetCurSel	101
8.12.8 GetCurText	101
8.12.9 RemoveItem	102
8.12.10 SetCurSel	102
8.13 日期选择控件 DateTimePicker 专用接口	103
8.13.1 GetDay	103
8.13.2 GetDayOfWeek	103
8.13.3 GetDayOfYear	104
8.13.4 GetYear	105
8.13.5 GetHour	105
8.13.6 GetMinute	106
8.13.7 GetSecond	106
8.13.8 GetMonth	107
8.14 树形控件 WebTreeView 专用接口	108
8.14.1 树形控件 WebTreeView 的事件函数 OnClick	108

8.14.2 AddNode	108
8.14.3 CheckAll	109
8.14.4 DeleteAllNodes	109
8.14.5 DeleteNode	110
8.14.6 GetCheckedLeafNodes	110
8.14.7 GetCheckedNodes	111
8.14.8 Inverse	112
8.14.9 UpdateNode	112
8.15 数据库表控件 DataBase 专用接口	113
8.15.1 CreateListColumn	113
8.15.2 AddDataToListCtrl	114
8.15.3 OnExport	114
8.15.4 QueryByField	115
8.15.5 ClearListCtrl	115
8.15.6 ClearQueryCondition	116
8.15.7 QueryBySql	116
9 事件触发函数	117
9.1 OnLButtonUp	117
9.2 OnLButtonDown	117
9.3 OnLButtonDblClk	117
9.4 OnRButtonClk	117
9.5 OnMouseMove	118
9.6 OnMouseEnter	118
9.7 OnMouseLeave	118
9.8 OnTimer	118
9.9 onDataChange	119
9.10 OnClosePic	119
9.11 OnOpenPic	119
10 调度支持的脚本	121
10.1 时间计划的脚本	121
10.1.1 时间计划的接口函数	121
10.1.2 时间计划的事件函数 OnTimeOut	125
10.2 事件计划相关脚本	125
10.2.1 事件计划的接口函数	125
10.2.2 事件计划的事件函数	127
10.3 调度支持的事件	129
10.3.1 Initialize	129
10.3.2 Close	130

10.4 调度支持的 APP 函数	130
10.4.1 OnSwitchServer	130
10.4.2 OnSwitchTeam	131
10.4.3 OnUserLogin	131
10.5 注意事项	131
11 脚本应用举例	132
11.1 从第三方数据库读数据	132
11.1.1 背景	132
11.1.2 代码	133
11.1.3 结果	135
11.2 写数据到第三方数据库	136
11.2.1 背景	136
11.2.2 代码	137
11.2.3 结果	138
11.3 全局脚本应用举例	138
11.3.1 背景	138
11.3.2 操作步骤	138
11.3.3 结果	139
12 资料版本说明	140
13 附录	141
13.1 颜色常数	141
13.2 背景风格对照表	141
13.3 边框风格对照表	142
13.4 渐变风格对照表	142
13.5 按钮风格对照表	143
13.6 MsgBox 中按钮设置和返回值说明	143
13.7 报警统计条件字符串	144
13.8 报警类型与子条件对照表	145

脚本编辑器

1 概述

1.1 脚本语言简介

脚本编辑器（VFScript.exe）采用 VBScript 脚本语言。

VBScript 是 Visual Basic 的一个子集，它最大优点在于简单易学。VBScript 仅保留了 Visual Basic 中少量关键字，大大简化了 Visual Basic 的语法，使得 VBScript 易学易用，即使是一个对编程语言毫无经验的人也可以在短时间内掌握 VBScript。

此外，VBScript 还具有安全性高、可移植性强等优点。脚本编辑器 VFScript 将灵活的脚本应用于监控软件中，能实现更强大的功能，是流程图绘制软件（VFDraw）、调度软件（VFSchedule）、组态管理软件（VxExplorer）的独特组成部分。

VFScript 的全部功能只有在 VFDraw、VFSchedule 或 VxExplorer 启动时才能使用。脚本文件的保存后缀是.vbs。一个流程图画面对应一个同名的.vbs 文件。一个调度对应一个同名的.vbs 文件。自定义结构类型的一个行为对应一个名为“PEM_结构类型名称_行为名称”的.vbs 文件。

1.2 功能特点

脚本编辑器软件具有以下特点：

- 使用标准的 VBScript 脚本语言，并在此基础上增加了流程图操作的扩展接口。
- 内置于 InPlant SCADA 软件中，通常只能在 VFDraw、VFSchedule 和 VxExplorer 中使用。
- 支持语法加亮和函数在线提示。
- 支持事件向导。
- 支持 VBScript 语法检查。

2 主界面

脚本编辑器软件主要用于编辑脚本语言。如图 2-1 所示，脚本编辑器界面由标题栏、菜单栏、工具栏、编辑区、状态栏、及多个子窗口组成，下面将分别介绍它们的功能。



图 2-1 脚本编辑器界面

2.1 标题栏

如图 2-1 所示，标题栏中显示 VBScript（编辑器软件名称）- VFDRAW（编辑器服务的对象：流程图 VFDRAW 或调度 VFSchedule）- 编辑模式（编辑器运行状态：编辑模式或调试模式）- Demo7（流程图名称）。

标题栏右侧还有“最大化”、“最小化”和“关闭”3 个窗口按钮操作图标，用户可以通过这些图标进行窗口大小的设定。

2.2 菜单栏和工具栏一览

本章节主要介绍菜单栏/工具栏中的各功能选项。



菜单栏上显示有文件、编辑、调试、设置、视图和帮助六个菜单项，每个菜单项右边括号中的字母表示菜单项的快捷键方式，同时按“ALT+快捷字母”即可打开菜单项，显示其子菜单的内容。

工具栏中显示了常用的菜单命令，便于快捷操作。

菜单栏/工具栏中若选项呈灰色，则表示当下该脚本无法进行此项操作。

2.2.1 文件菜单

表 2-1 文件菜单

菜单栏	工具栏	说明
保存 (S)		将已完成的脚本程序保存在硬盘上
打印 (P)		用于打印所建立并已保存的脚本文件 注：不同的打印驱动在打印界面上会稍有不同

菜单栏	工具栏	说明
关闭并返回 (x)	-	用于关闭脚本编辑器软件，返回到流程图或调度界面

2.2.2 编辑菜单

编辑菜单中所有功能，还可以通过右键菜单实现。在脚本编辑区任意地方单击右键，在弹出的右键菜单中选择相应的功能即可。

表 2-2 编辑菜单

菜单栏	工具栏	说明
撤销 Ctrl+Z		支持用户在编辑脚本时通过撤消来恢复前面的操作
重复 Ctrl+Y		支持用户在编辑脚本时通过重复来取消前面的撤消操作
剪切 Ctrl+X		用于将脚本编辑区中用户选定区域的内容复制到剪切板内，同时删除该区域内的内容
复制 Ctrl+C		用于将脚本编辑区中用户选定区域的内容复制到剪切板内，不删除选定区域的内容
粘贴 Ctrl+V		用于将剪切板中的最新内容（即最近一次剪切或复制的内容）复制到指定的脚本编辑区
删除 Delete	-	用于将编辑区中用户选取的内容删除
选择全部 Ctrl+A	-	将脚本编辑区中的内容全部选中
查找 Ctrl+F	-	用于查找编辑区中指定的单词
查找下一个 F3	-	用于查找下一个相同的内容
替换 Ctrl+H		用于批量替换脚本文件的某个字段
锁定文本		用于将编辑区中的脚本锁定或解除锁定。锁定文本后，将不能进行编辑操作，可以避免因错误操作导致对脚本的不当修改
-		注释代码：对某些行内容设置为注释内容
-		取消注释代码：将某些行注释内容取消注释
-		用户自定义函数库：打开用户函数库界面，可自定义函数

2.2.3 调试菜单

表 2-3 调试菜单

菜单栏	工具栏	说明
编译		对当前脚本程序进行语法检查，检查结果显示在信息栏中
函数重名检测	-	对当前脚本程序进行函数重名检查，检查结果显示在信息栏中
逐语句 F11		逐语句调试
逐过程 F10		逐过程调试
跳出 SHIFT+F11		跳出调试
运行 F5		运行或继续调试

菜单栏	工具栏	说明
停止 SHIFT+F5		停止调试
添加/删除断点 F9		选中某行脚本，对该行进行添加或者删除断点
删除所有断点 CTRL+SHIFT+F9		删除当前脚本内的所有断点
使能/不使能断点 F6		改变当前选中行断点使能状态
不使能所有断点 CTRL+SHIFT+F6		当前脚本内的所有断点不使能

2.2.4 设置菜单

表 2-4 设置菜单

菜单栏	说明
编辑环境选项...	用于对脚本编辑区的背景、文本、关键字加亮、注释和数字的颜色进行设置，并提供预览功能
永远置顶	当前脚本编辑界面将被置顶，即无法将其它界面打开为当前活动界面

2.2.5 视图菜单

视图菜单下可显示或隐藏各工具条。选中视图菜单中的工具栏名称即可在编辑界面中显示此工具栏；反之，则隐藏此工具栏。

表 2-5 视图菜单

菜单栏	工具栏	说明
系统工具栏 (T)	-	显示/隐藏系统工具栏
编辑工具栏 (E)	-	显示/隐藏编辑工具栏
调试工具栏 (D)	-	显示/隐藏调试工具栏
状态栏 (S)	-	显示/隐藏状态栏
输出窗口 (O)	-	显示/隐藏输出窗口
变量窗口 (V)	-	显示/隐藏变量窗口
堆栈窗口 (A)	-	显示/隐藏堆栈窗口
表达式窗口 (X)	-	显示/隐藏表达式窗口
对象浏览器窗口 (J)	-	显示/隐藏对象浏览器窗口
工程窗口 (P)	-	显示/隐藏工程窗口
函数库窗口 (F)	-	显示/隐藏函数库窗口

2.2.6 帮助菜单

帮助菜单提供了有关 VFScript 软件的版权及使用权的相关信息。

表 2-6 帮助菜单

菜单栏	说明
-----	----

菜单栏	说明
帮助 F1	在线帮助链接
关于 (A)	显示脚本编辑器版本信息

2.3 右键菜单一览

右键菜单：在主界面不同区域右键单击后弹出的菜单命令。本小节主要介绍不同窗口的右键菜单功能。

表 2-7 工程窗口右键菜单

菜单栏	说明
添加函数	弹出“添加函数”界面
事件向导	选中某对象时的右键菜单，弹出“添加事件函数”界面

表 2-8 脚本编辑区右键菜单

菜单栏	说明
跳转至 CTRL+G	弹出“跳转至”界面，可跳转到指定行
显示行号	显示/隐藏脚本行号
撤销~锁定文本	与菜单栏“编辑”下的命令项功能一致，不再重复说明

表 2-9 输出窗口右键菜单

菜单栏	说明
复制消息	复制所有输出信息
清空消息	清空所有输出信息
选择第一条	选择第一条输出信息
选择最后一条	选择最后一条输出信息

表 2-10 对象浏览器菜单

菜单栏	说明
全选	注释框中的右键菜单，全选注释框中的内容
从右到左阅读顺序	注释框中的右键菜单，使注释框中的内容从右到左阅读

表 2-11 函数库菜单

菜单栏	说明
导出函数库	树形列表中的右键菜单，导出自定义函数库
刷新	树形列表中的右键菜单，刷新函数库
全选	注释框中的右键菜单，全选注释框中的内容

菜单栏	说明
从右到左阅读顺序	注释框中的右键菜单，使注释框中的内容从右到左阅读

2.4 工程窗口

工程窗口：以树型结构显示流程图中的所有对象和脚本中已有的对象相关事件。

工程窗口默认位于脚本编辑区左侧，如图 2-2 工程窗口所示。

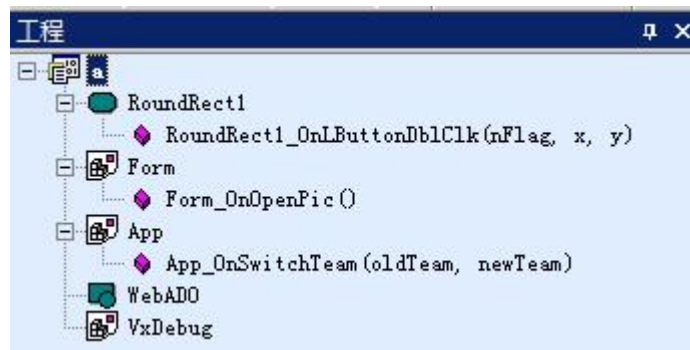


图 2-2 工程窗口

2.5 输出/变量/堆栈/表达式窗口

输出窗口：显示编译信息、调试信息和查找信息。

表达式窗口：显示全部过程中的表达式的值。

变量窗口：显示全部过程中的变量值。修改变量值需满足以下规则

- 数值类型的变量，直接输入数字。
- 字符串类型的变量，必须使用引号，如："hello"。
- 允许输入变量名，使被修改变量的值等于输入变量的值（如果输入不存在的变量名，则会新增该变量）。

堆栈窗口：支持手动选择不同层的堆栈，并自动切换到相应堆栈的上下文环境进行显示、调试。

四个窗口默认位于脚本编辑区的中下方，且以 Tab 页形式显示在同一区域内，如图 2-3 所示。

根据实际需求，可将每个窗口调整为独立窗口和自动隐藏。具体操作说明可查看 22.7 拖动和自动隐藏。Tab 页顺序可调整。



图 2-3 输出/变量/堆栈/表达式窗口

2.6 对象浏览器/函数库窗口

对象浏览器和函数库窗口默认位于界面右侧，且以 Tab 页形式显示在同一区域内，如图 2-4 所示。根据实际需求，可将每个窗口调整为独立窗口和自动隐藏。具体操作说明可查看 2.7 拖动和自动隐藏。Tab 页顺序可调整。

对象浏览器窗口

对象浏览器窗口用于显示该流程图内含有的对象函数及用法说明，帮助用户更快的了解对象函数应用，用法说明可直接复制到脚本编辑区使用。

窗口如图 2-4 所示，上部分以树形结构列举每个对象的函数，对象函数前的字母 P 表示该函数为属性类函数，M 表示为方法类函数。下部分注释框中显示该函数的使用说明。函数应用说明可参看窗体对象函数～事件触发函数章节内容。

流程图内新增一个对象，则此窗口内将自动添加该对象的对应函数和用法说明。

固定显示的对象函数中 VxDebugLib 对应脚本调试信息输出函数，ScriptAPILib 对应 APP 对象，SCView 对应 Form 对象。

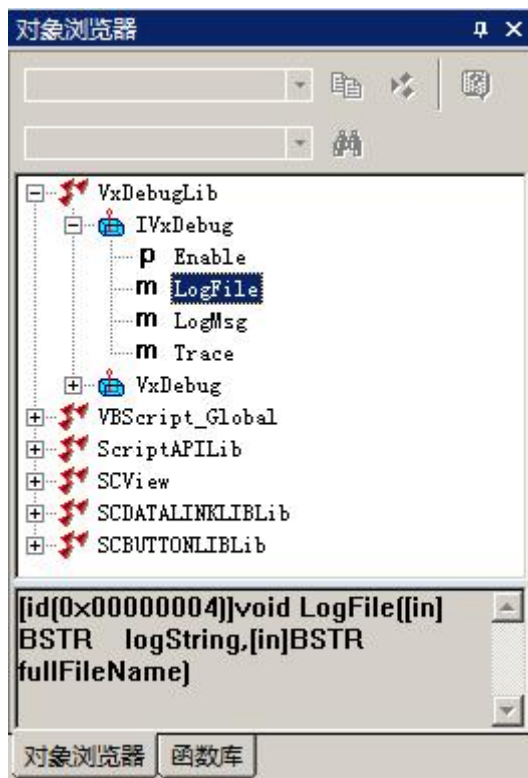


图 2-4 对象浏览器窗口

函数库窗口

函数库窗口用于显示内置脚本函数库和用户自定义函数库。窗口如图 2-5 所示，上部分以树形结构显示函数库信息，下部分注释框中对选中的函数参数进行说明。双击函数名可将该函数添加到代码中。


用户可通过工具栏  按钮打开用户自定义函数库编辑界面，新增自定义函数，具体操作可查看自定义函数。



图 2-5 函数库窗口

2.7 拖动和自动隐藏窗口

拖动窗口

将窗口从默认位置拖动到界面其他位置。

1. 选中并长按窗口标题栏，将其拖动到界面任意处。
2. 拖动时界面中出现如图 2-6 所示的工具，将图拖至工具不同方向上，即可预览窗口调整位置。
3. 放开鼠标，窗口将自动调整至相应位置。

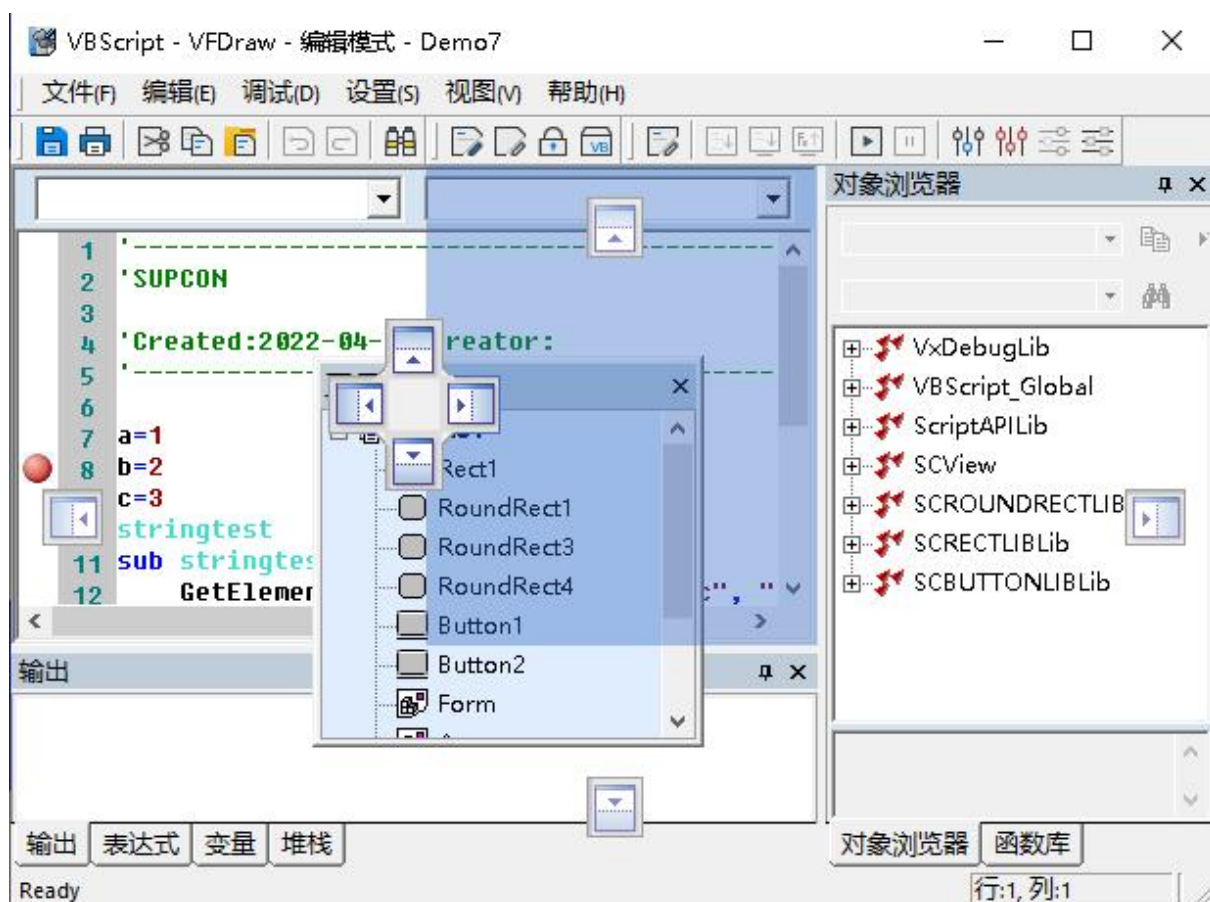



图 2-6 拖动窗口

自动隐藏

鼠标移至主界面边缘窗口名称处则显示该窗口，否则呈隐藏状态。

在脚本编辑器中，每个可设置显示/隐藏的窗口都支持自动隐藏功能，点击窗口右上角的图标，即可将窗口隐藏在就近的界面边缘。如下图所示，为位于界面左侧的“工程”窗口隐藏后的状态。

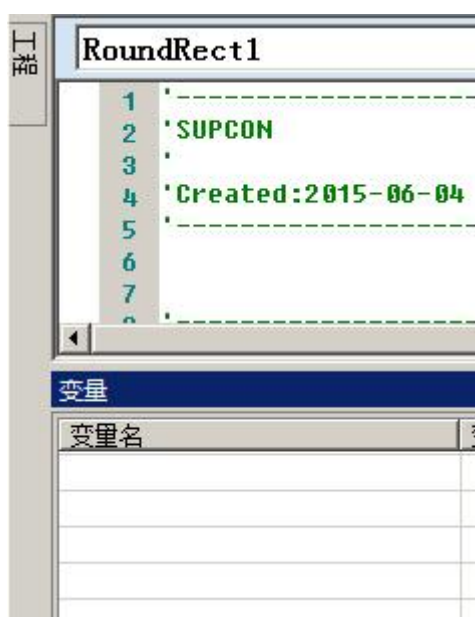


图 2-7 自动隐藏窗口

3 基本操作

本章节描述脚本文件的创建、调试、保存、查找、替换、复制粘贴等等一些基本操作。

3.1 创建脚本

在流程图绘制软件中，选择菜单命令【工具/编辑脚本】，或在流程图绘制区的空白处单击右键并在右键菜单中选择“编辑脚本”选项启动流程图脚本编辑器软件。

在调度软件中，点击“编辑脚本”按钮后，在弹出的脚本编辑器中编辑脚本并保存。

语法检查通过后，选择菜单命令【文件/保存】，或单击按钮保存文件。

脚本文件与对应的流程图或调度文件名一致，仅后缀不同。若某流程图或调度文件名与其对应的脚本文件名不一致，则脚本无法运行。

3.2 查找内容

脚本编辑器提供查找功能，可查找脚本文件中的代码单词、脚本注释等。

左键单击【编辑/查找】，弹出如图 3-1 所示的查找对话框。在查找栏中输入要查找的内容，点击“查找”即可。

可对查找的单词进行以下定位：

- 匹配整个单词：表示查找结果和查找内容是同一个单词。
- 大小写匹配：表示查找结果和查找内容是同一个单词，且大小写相同。

查找方向：

- 向上：表示从光标所在位置向上查找单词。
- 向下：表示从光标所在位置向下查找单词。




图 3-1 查找对话框

在查找到一处匹配的内容后，若需要继续查找同一内容，则可左键单击【编辑/查找下一个】或快捷键 F3 即可。

3.3 替换内容

脚本编辑器提供替换功能，可批量替换脚本文件的某个字段。

左键单击【编辑/替换】或按钮，弹出如图 3-2 所示界面，在查找栏中输入被替换的内容，在替换栏中输入替换的内容，点击“查找下一个”，查找到准确内容后点击“替换”即可。

在编辑区选择部分内容，再进行替换功能时，替换范围中“被选择部分”可选，否则只能全文替换。

点击“替换全部”即可替换所有替换范围内匹配的内容。

点击“取消”即退出替换功能。

“匹配整个单词”和“大小写匹配”同查找内容。



图 3-2 替换对话框

3.4 剪切、复制和粘贴脚本

脚本编辑器提供剪切、复制、粘贴的功能，对同一脚本文件中和不同脚本文件间的操作都有效。可通过“编辑”菜单、右键菜单或快捷键的形式来实现以上功能。

在编辑区选定需要剪切或复制的内容，选择剪切或复制选项，即将内容剪切或复制到剪贴板中，左键单击【编辑/粘贴】即可将剪贴板中的内容粘贴到选定的位置上。





提示：

复制/粘贴支持在不同脚本文件之间的操作，复制后可进行多次粘贴。

3.5 撤销更改

撤销即取消用户进行的操作，从最新的操作往前撤销。对脚本文件打开后当下进行的操作有效，对文件上一次打开时所作的操作无效。


重复即还原被撤销的内容，从最新撤销的内容往前重复。在进行撤销操作后未对脚本进行其他编辑时有效。

工具栏和按钮，右键菜单撤销和重复选项，或者菜单栏“编辑”下的撤销和重复选项，都可以实现脚本的撤销和重复功能。

撤销或重复选项变灰时，表示无法进行撤销或重复操作。

3.6 编辑环境设置

编辑环境设置包括对编辑区背景色、文本色、关键字颜色、注释颜色、数字颜色、函数颜色和字体进行设置。选择菜单栏【设置/编辑环境选项】，即可打开设置界面。

颜色设置：在设置界面左边选择设置项后，单击颜色一栏，如图 3-3 所示，当光标变成试管形状时可提取界面其他颜色或直接在弹出的颜色栏中选取颜色。设置的色彩效果可在预览框中查看。点击颜色栏右下角可以选择更多的颜色。

字体设置：针对整个脚本文件中的所有文字，在界面左侧选择“字体设置”后，在右侧将字体设置为常规、粗体、或斜体。设置效果可在预览框中查看。



图 3-3 编辑环境设置

恢复全局默认：所有设置恢复默认设置。

3.7 编程助手

编程助手可快速显示适用当前图形对象、窗体对象以及 App 对象等的可编辑属性，如图 3-4 所示（以图形对象矩形为例）。

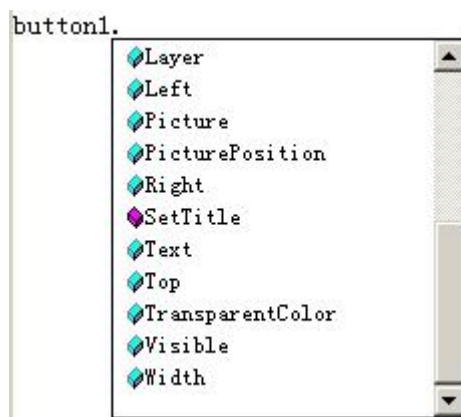


图 3-4 矩形对象的属性

操作方法：输入前缀后，在半角英语输入法状态下输入“.”。

图形对象的前缀为图形对象名称，窗体对象的前缀为 Form，App 对象的前缀为 App。

编程助手中有红色标识和蓝色标识的区别：

- 蓝色标识指的是图形对象属性，是对对象特征的描述，比如报表的名字、宽度等等（即公共接口）。
- 红色标识指的是方法，可以看作要达到目的一种手段。

函数的使用方法请查阅章节窗体对象函数～事件触发函数章节内容。

3.8 AI 编程助手

在 VBS 脚本编辑器界面，可调用 AI 编程小助手，在小助手界面对小助手进行提问，帮助快速完成组态 VBS 脚本编程。



图 3-5 工具栏 AI 图标

操作方法：在 VBS 脚本编辑器界面，AI 编程小助手可通过双击 ALT 快捷键或右键单击工具栏的 AI 键弹出小助手，如果触发快捷键未弹出小助手时，打开 Windows 安全中心 - 病毒威胁和防护 - 保护历史 找到保护历史记录 -> 已隔离的威胁 -> 点击操作 -> 还原，然后再触发快捷键。

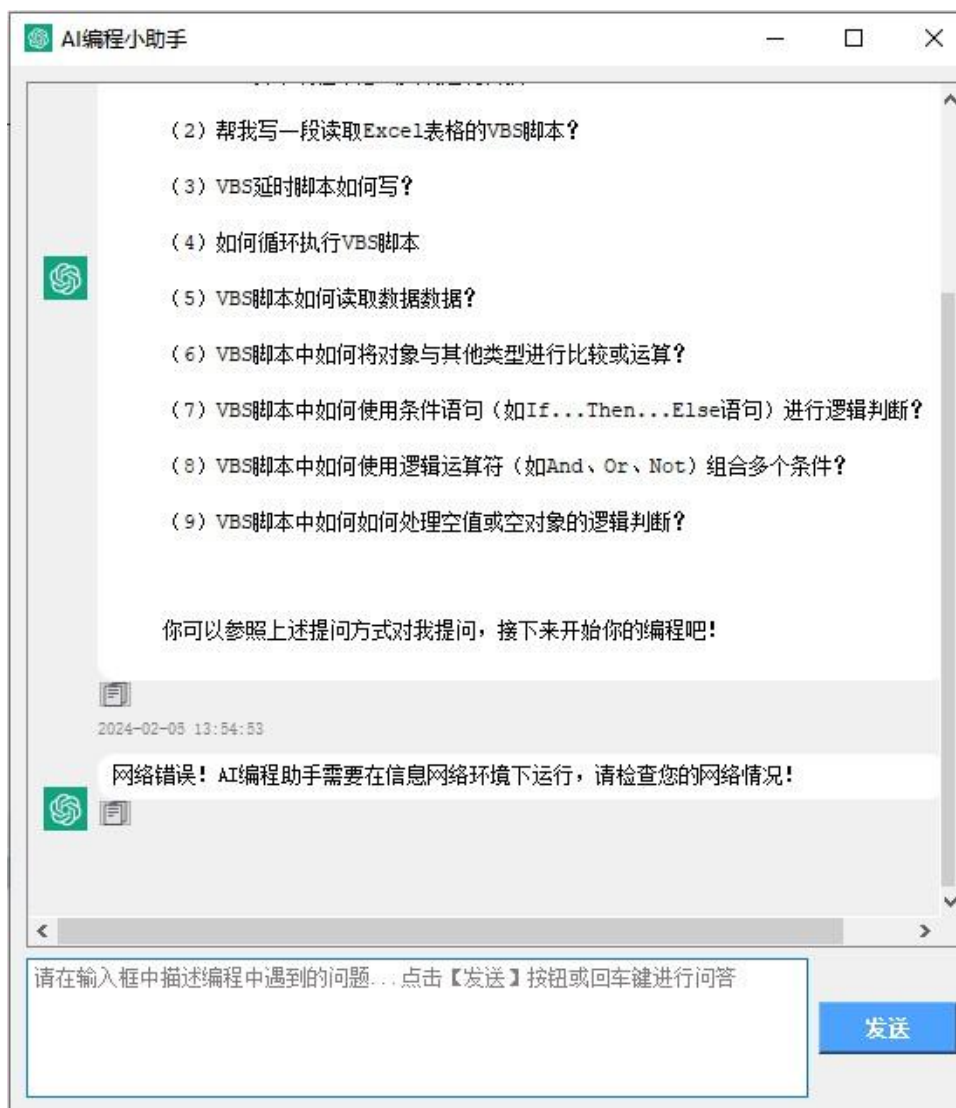


图 3-6 AI 编程小助手

3.9 位号选择器功能

编辑位号及相关信息时，可使用右键中的位号选择器功能快速填入位号名或者位号相关属性。



图 3-7 右键菜单

在位号选择器中可设置过滤条件搜索位号，包括“数据库位号”和“系统位号”，筛选结果在位号名列表中显示，在位号名列表中选中位号，点击“确定”即可添加位号到脚本中。

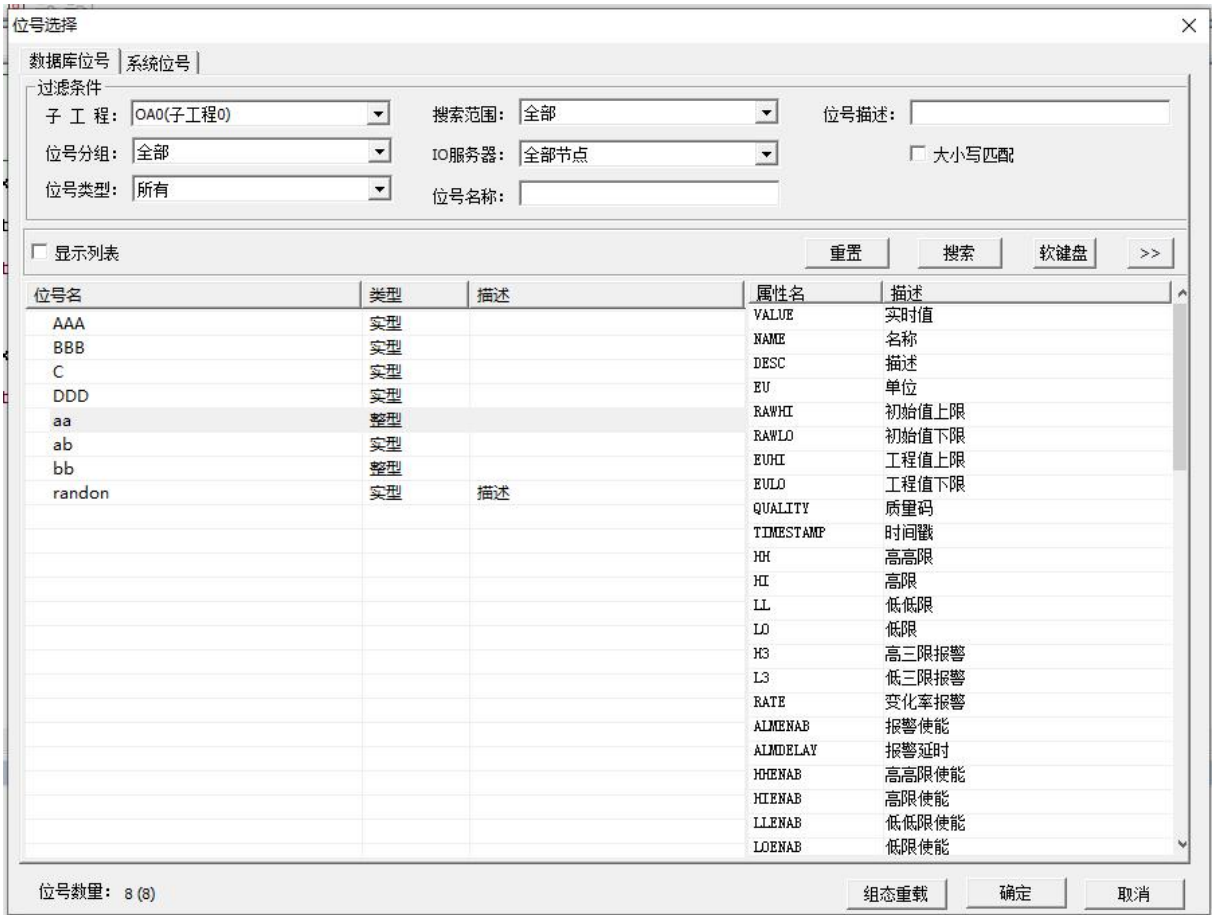


图 3-8 位号选择器

3.10 添加函数和事件向导

为了方便创建函数和事件，脚本编辑器提供了“添加函数”和“事件向导”两项功能。

选中脚本信息栏中的任一图形对象名鼠标右键单击该图形对象名，其右键菜单中均包含“添加函数”和“事件向导”菜单选项。选择“添加函数”，弹出“添加函数”对话框，如图 3-7 所示。其中 Function 函数可以使用返回值，Sub 函数没有返回值。

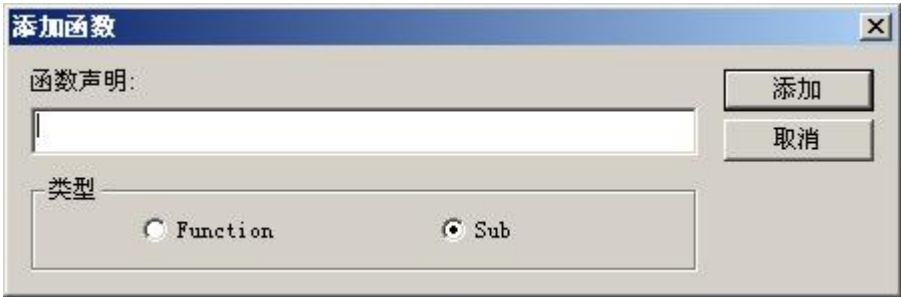


图 3-9 函数添加向导

选择“事件向导”，弹出“添加事件函数”对话框，如图 3-8 所示。选择“对象”和“事件”后，点击“添加”，在脚本中将会自动增加事件函数的定义。



图 3-10 事件向导

3.11 事件向导工具栏



事件向导工具栏位于脚本编辑区上方，可快速添加对象事件，方便用户使用。

如图 3-9 所示，事件向导工具栏包括“对象选择框”和“事件选择框”两部分。“对象选择框”可显示当前流程图或调度中所有对象的英文名称；“事件选择框”内显示各对象所支持的事件。



图 3-11 事件向导工具栏


操作步骤

1. 鼠标左键双击“脚本信息栏”中的任一对象名，“对象选择框”中将自动写入该图形对象名，或者在“对象选择框”中，单击 ，在下拉框中选择对象名。
2. 在“事件选择框”中，单击 ，选择某事件触发类型，该触发类型将自动被写入到编辑工作区中，形成如下所示的原始脚本文本：

```
Sub Button5_OnLButtonUp(nFlag, x, y)
```

```
End Sub
```

3.12 自定义函数

点击工具栏  按钮，打开自定义函数编辑界面，如图 3-10 所示，在右侧函数库中，右击“自

定义函数库”，右键菜单功能介绍如下：

- 添加函数：添加一个函数，并在代码编辑区添加函数框。
- 删除函数：删除选中的函数。
- 添加分组：添加一个分组，可将同类函数分组存放。
- 编辑分组：重命名分组。
- 删除分组：删除选中的分组。请先删除分组下的函数再删除分组，否则无法删除。
- 导入函数库：导入.xml 格式的函数库。
- 导出函数库：可导出部分或全部自定义函数，文件格式.xml。

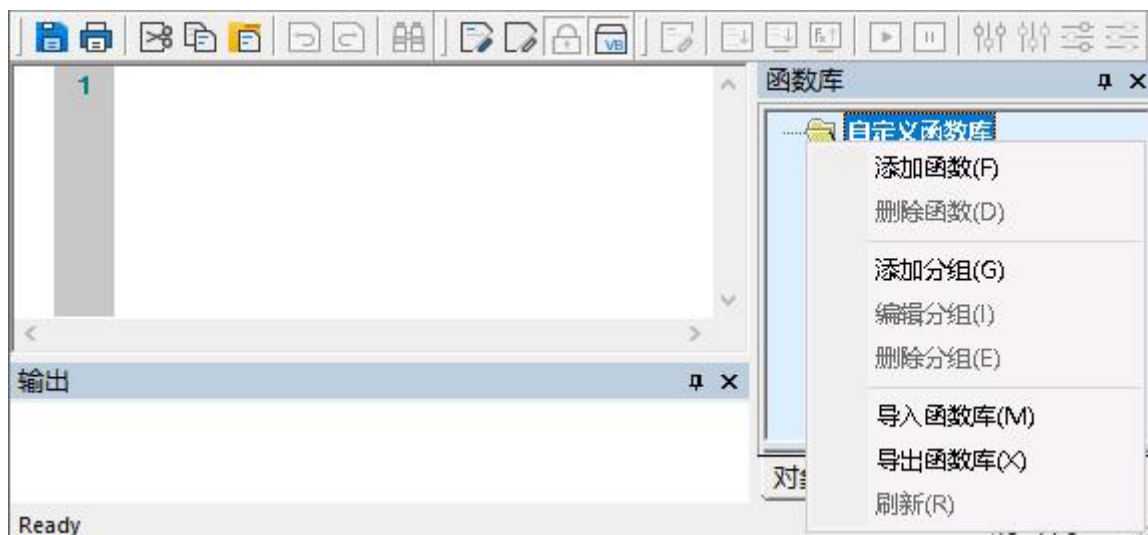


图 3-12 添加函数

举例：如需自定义 2 个加法函数（三个数加法和四个数加法），可对这类加法函数进行分组管理，操作步骤如下：

- 1) 选中“自定义函数库”，在右键菜单中选择“添加分组”，命名为“加法函数”。



图 3-13 建分组

- 2) 选中“加法函数”，在右键菜单中选择“添加函数”，代码编辑如图 3-14 所示。以相同方式再新增一个函数。

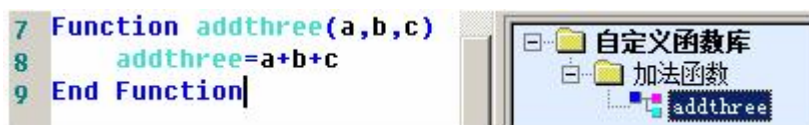


图 3-14 新增自定义函数 1

- 3) 如图 3-13 所示，新增另一个加法函数。

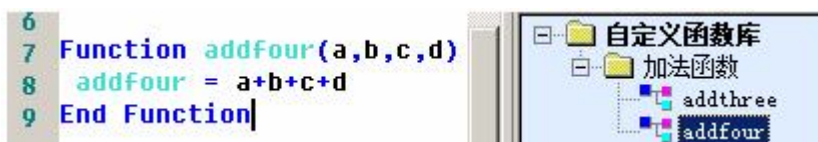


图 3-15 新增自定义函数 2

4) 自定义函数编辑完成后，点击 按钮退出函数编辑界面，返回脚本编辑器界面，自定义函数库如图 3-14 所示，用户可直接引用。



图 3-16 自定义函数库

3.13 脚本调试说明

InPlant SCADA 脚本调试功能支持断点调试、单步调试等等，调试信息可在“输出窗口”中查看，调试过程中的表达式、变量、堆栈等信息可在“表达式/变量/堆栈窗口”中查看。



提示：

若脚本中使用了 **app**, **form**, **vxdebug**, 以及流程图控件对象（如 **button** 等），需要通过流程图在线调试来触发调试入口，否则脚本调试期将不识别此类对象。

含义

- 断点调试：调试时，程序运行到设置断点的代码行后将暂时停止。
- 逐语句调试：当脚本运行到的某一行脚本中含有子过程调用，脚本会执行到该子过程的第一行有效代码处停留下来。
- 逐过程调试：如果此行脚本中有自定义的子过程调用且子过程内部有断点时，脚本会直接执行到此子过程的断点处，否则脚本会运行到下一句。
- 跳出调试：当脚本执行停留在某个子过程内部时，运行跳出调试，将会跳出该子过程，停留在该子过程调用行的下一行脚本处。

操作方法

1. 通过点击工具栏 按钮，或者快捷键 F9，在需要观察的代码前设置断点，如图 3-15 所示。



图 3-17 设置断点






2. 点击工具栏  按钮，或者快捷键 F5，使程序进入调试状态，如图 3-16 所示。



图 3-18 调试

3. 当程序运行到断点行代码时将暂时停止，此时用户可以根据需要进行如下操作：
- 逐句调试 ( 或 F11)。
 - 逐过程调试 ( 或 F10)。
 - 跳出调试 ( 或 SHIFT+F11)。
 - 点击  按钮继续调试，若后续代码有断点，则将直接执行到下一个断点处暂停。
4. 调试时，可在变量窗口中查看变量值，如图 3-17 所示。

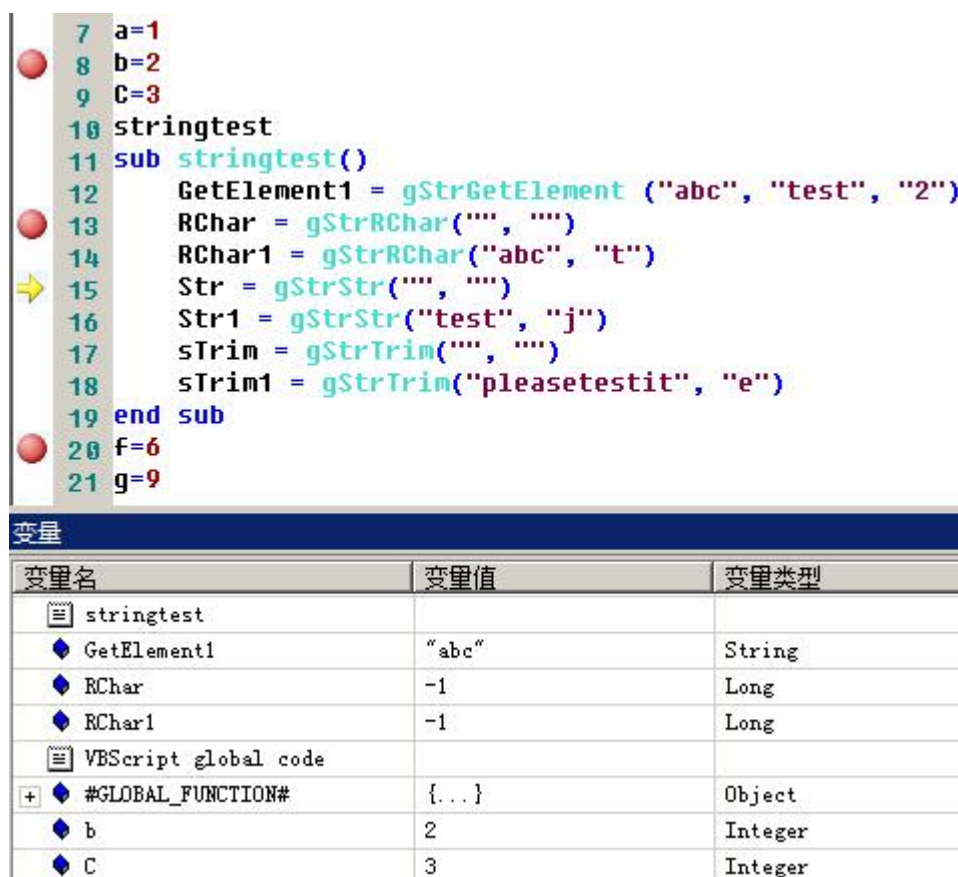


图 3-19 查看变量

5. 点击 按钮停止调试。

4 脚本引用类型

不同类型的功能可引用不同类型的脚本，详细分类说明如下：

- 流程图：可引用 Form、App、WebADO、脚本调试函数（以 VxDebug 开头）和简单图形对象的脚本。
- 图符：可引用 Form（原有 Form 对象的 ShutDown、Tag 函数）、App（原有 APP 对象的 ReadValue、SaveValue、GetCurrentUser、GetCurrentUserGrade、Mute、GetGroup 函数）、WebADO、脚本调试函数（以 VxDebug 开头）、已添加的简单图形对象和模板对象的脚本。
- 面板：可引用 Form（原有 Form 对象的 ShutDown、Tag 函数）、App（原有 APP 对象的 ReadValue、SaveValue、GetCurrentUser、GetCurrentUserGrade、Mute、GetGroup 函数）、WebADO、脚本调试函数（以 VxDebug 开头）、已添加的简单图形对象、已添加的一般控件的脚本。
- 调度：可引用 Scheduler、脚本调试函数（以 VxDebug 开头）和 App 的脚本。
- 自定义结构的行为：可引用 App 脚本。

5 脚本调试信息输出函数

利用 Vxdebug 可以将脚本调试信息输出到标准 API 信息窗口、操作日志或者指定的文本文件中。当脚本调试信息输出到标准 API 信息窗口时，可通过 DebugView 工具获取并查看。

脚本调试函数在脚本中默认为“VxDebug”开头的函数。

5.1 Enable

Enable 函数用于使能调试信息的输出。

语法

```
vxdebug.Enable = 0 或 1
```

输入参数

0 或 1, 0 表示不输出调试信息, 当某信息不输出时添加这行代码; 1 表示输出调试信息, 脚本系统中默认输出, 因此无需特别添加此代码。

返回值

无

举例

当某信息不输出时添加代码 “VxDebug.Enable=0”。

当某信息需要输出时添加代码 “VxDebug.Enable=1”。

5.2 Trace

Trace 函数用于将指定的调试信息输出到标准的 API 信息窗口。

语法

```
vxdebug.Trace(msg)
```

输入参数

msg, 字符串形式, 输出到标准 API 信息窗口的调试信息。

返回值

无

举例

单击 Button1, 在标准 API 窗口中输出 “[VxDebug]Button Clicked”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    vxdebug.Trace "[VxDebug]Button Clicked"
End Sub
```

5.3 LogMsg

LogMsg 函数用于将指定的调试信息输出到操作日志。

语法

```
vxdebug.LogMsg(msg)
```

输入参数

msg, 字符串形式, 输出到操作日志的调试信息。

返回值

无

举例

单击 Button1, 在操作日志中输出 “[VxDebug]Button Clicked”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    vxdebug.LogMsg"[VxDebug]Button Clicked"
End Sub
```

5.4 LogFile

LogFile 函数用于将指定的调试信息输出到指定的文本。

语法

```
vxdebug.LogFile (msg,filename)
```

输入参数

msg, 字符串形式, 输出到操作日志的调试信息。

filename, 字符串形式, 文本文件的路径及文件名。

返回值

无

举例

单击 Button1, 在 D 盘 Logfile.txt 中输出 “[VxDebug]Button Clicked”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    vxdebug.LogFile"[VxDebug]Button Clicked"," D:\Logfile.txt "
End Sub
```

6 窗体对象函数

窗体对象函数即流程图画面专属的函数，一副流程图画面对应一个窗口。在编辑脚本过程中，窗口写为“Form”。通过窗口函数，可以获取窗口中所有的报警信息、指定位号的属性以及弹出画面等操作。



提示：

函数是返回一个值的运算；属性指可以改变的图形对象属性；事件指触发一个过程的执行。

6.1 AccessLevel

AccessLevel 函数用于获取窗口中指定用户的权限等级。

语法

```
Form.AccessLevel(userName)
```

参数

UserName，当为"*"时，表示获取当前用户的权限等级；当为指定用户名时，表示获取指定用户的权限等级。

返回值

返回值为 0~9。

其中，9 表示特权+，8 表示特权，7 表示特权-，6 表示工程师+，5 表示工程师，4 表示工程师-，3 表示操作员+，2 表示操作员，1 表示操作员-，0 表示观察员。另外，-1 表示用户不存在。

举例

在流程图中添加按钮 Button7 和 Button8，单击 Button8 弹出提示框显示 admin 的用户等级，单击 Button7 显示当前登录用户的用户等级。

```
Sub Button8_OnLButtonUp(nFlag, x, y)
    Dim A
    A=Form.AccessLevel("admin") '获取用户 admin 的权限等级，并赋值给 A
    app.MsgBox A, "提示框", 0 '通过提示框显示 A 的内容
End Sub
Sub Button7_OnLButtonUp(nFlag, x, y)
    DIM A
    A=Form.AccessLevel("*") '获取当前用户权限等级，并赋值给 A
    app.MsgBox A, "提示框", 0
End Sub
```

代码生效后，单击按钮 Button8 弹出如图 6-1 所示左侧的提示框，表示 admin 等级为 9，即为最高等级特权+。单击按钮 Button7 弹出如图 6-1 所示右侧的提示框，表示当前登录用户等级为 0，即为观察员权限。



图 6-1 信息框

6.2 Alarm

Alarm 函数用于获取窗口中指定位号的报警状态。

语法

```
Form.Alarm(tagName)
```

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“THIS.LIMIT”中“THIS”为位号名、“LIMIT”为字段。

返回值

返回值为 True 或 False。True 表示位号有报警, False 表示位号无报警。

举例

在流程图中添加按钮 Button2, 单击按钮, 将在 D 盘的 Logfile.txt 显示“报警状态:False”的信息。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
    D = Form.Alarm("A")
    VxDebug.LogFile "报警状态:" & D, "D:\Logfile.txt"
End Sub
```

6.3 ClosePopPic

ClosePopPic 函数用于关闭弹出式流程图。

语法

```
Form.ClosePopPic(lpszPicName)
```

参数

lpszPicName: 流程图名, 字符串形式。

若当前画面为弹出式流程图画面, 则关闭当前弹出式流程图, lpszPicName 指定无效。

若当前画面为普通流程图画面, 则关闭 lpszPicName 指定的弹出式流程图。

举例

在弹出式流程图“流程图.pic”中添加按钮 Button3，单击 Button3 将关闭该弹出式流程图。

```
Sub Button3_OnLButtonUp(nFlag, x, y)
    Form.ClosePopPic("流程图.pic") '输入流程图名称
End Sub
```

相关函数

OpenPic

6.4 GetTagOffColor

GetTagOffColor 函数用于获取指定开关量位号 OFF 时的颜色。

语法

```
Form.GetTagOffColor(tagName)
```

参数

tagName，位号名，采用字符串形式，支持结构位号。结构位号的格式为“位号名.位号的字段”，如“A1.Value”中“A1”为位号名、“Value”为字段。

返回值

位号的 OFF 颜色。

举例

在流程图添加按钮 Button6，单击 Button6 将弹出提示框显示开关量位号 C 的 OFF 颜色。弹出信息框显示数值 65280，根据表 13-1 所示，C 置 Off 时的颜色为绿色。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
    A=Form.GetTagOffColor("C") '获取开关量位号 C 置 Off 时的颜色并赋值给 A
    app.MsgBox A, "提示框", 0
End Sub
```

相关函数

GetTagOffDesc、GetTagOnColor、GetTagOnDesc

6.5 GetTagOffDesc

GetTagOffDesc 函数用于获取指定开关量位号 OFF 时的描述。

语法

```
Form.GetTagOffDesc(tagName)
```

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“A1.Value”中“A1”为位号名、“Value”为字段。

返回值

位号的 OFF 描述信息, 字符串形式。

举例

在流程图中添加按钮 Button6, 单击按钮 Button6 将获取开关量 D 的 off 描述并将其写入到 LogfileA.txt 文件中, 将开关量 D 的 ON 描述写入到 LogfileB.txt 文件中。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
    A=Form.GetTagOffDesc("D")
    '获取开关量位号 D 置 OFF 时的描述并赋值给 A
    B=Form.GetTagOnDesc("D")
    '获取开关量位号 D 置 ON 时的描述并赋值给 B
    VxDebug.LogFile A, "D:\LogfileA.txt" '将描述写入 D 盘 LogfileA.txt 文件中
    VxDebug.LogFile B, "D:\LogfileB.txt" '将描述写入 D 盘 LogfileB.txt 文件中
End Sub
```

相关函数

GetTagOffColor、GetTagOnColor、GetTagOnDesc

6.6 GetTagOnColor

GetTagOnColor 函数用于获取指定开关量位号 ON 时的颜色。

语法

Form.GetTagOnColor(tagName)

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“A1.Value”中“A1”为位号名、“Value”为字段。

返回值

位号的 On 颜色。

举例

在流程图中添加按钮 Button6, 单击 Button6 将弹出提示框显示开关量位号 c 的 ON 颜色。若 c 的 ON 颜色为红色则提示框中显示文字“红色”, 若 c 的 ON 颜色为绿色则提示框中显示文字“绿色”, 若 c 的 ON 颜色为其他颜色则直接显示颜色值。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
    A=Form.GetTagOnColor("c")
```

```

If A= 65280 Then
    app.MsgBox "绿色", "提示框", 0
    '如果 A 为 65280 则弹出信息框提示为“绿色”，值与颜色的对应关系请参见表 13-1
ElseIf A = 255 Then
    app.MsgBox "红色", "提示框", 0      '如果 A 为 255 则弹出信息框提示为“红色”
Else
    app.MsgBox A, "提示框", 0          '如果都不是，则直接显示 A 的值
End If
End Sub

```

相关函数

GetTagOffColor、GetTagOffDesc、GetTagOnDesc

6.7 GetTagOnDesc

GetTagOnDesc 函数用于获取指定开关量位号 ON 时的描述。

语法

```
Form.GetTagOnDesc(tagName)
```

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“A1.Value”中“A1”为位号名、“Value”为字段。

返回值

位号的 On 描述信息, 字符串形式。

举例

在流程图添加按钮 Button6, 单击按钮 Button6 将获取开关量 D 的 off 描述并将其写入到 LogfileA.txt 文件中, 将开关量 D 的 ON 描述写入到 LogfileB.txt 文件中。

```

Sub Button6_OnLButtonUp(nFlag, x, y)
    A=Form.GetTagOffDesc("D")
    '获取开关量位号 D 置 OFF 时的描述并赋值给 A
    B=Form.GetTagOnDesc("D")
    '获取开关量位号 D 置 ON 时的描述并赋值给 B
    VxDebug.LogFile A, "D:\LogfileA.txt" '将描述写入 D 盘 LogfileA.txt 文件中
    VxDebug.LogFile B, "D:\LogfileB.txt" '将描述写入 D 盘 LogfileB.txt 文件中
End Sub

```

相关函数

GetTagOffColor、GetTagOnColor、GetTagOffDesc

6.8 IsDomainTag

IsDomainTag 函数用于判断指定位号是否为点域位号。

语法

Form.IsDomainTag(tagName)

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“A1.Value”中“A1”为位号名、“Value”为字段。

返回值

1 表示指定位号存在且为点域位号, 0 表示位号存在且不是点域位号, -1 表示位号不存在。

举例

在流程图中添加按钮 Button2, 单击该按钮判断位号 A 是否为点域。若为点域则对其赋值 100, 若非点域则弹出信息框提示“不是点域, 无法对其进行写值”。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
Dim domainTag
    domainTag = Form.IsDomainTag("A") '将判断返回值赋值给 domainTag
    If domainTag = 0 Then
        App.MsgBox "不是点域, 无法对其进行写值", " SCADA ", 0
    Else
        App.WriteTag "A", 100
    End If
End Sub
```

6.9 Login

Login 函数用于弹出登录窗口, 并可以在登录窗口中切换用户。

语法

Form.Login

举例

在流程图中添加按钮 Button3, 单击 Button3 将弹出对话框“是否登录到监控系统? ”。单击对话框中的“确定”进入登录界面, 否则取消登录。

```
Sub Button3_OnLButtonUp(nFlag, x, y)
Dim needLogin
    needLogin = App.MsgBox ("是否登录到监控系统? ", " SCADA ", 1)
    If needLogin=1 Then          '信息框中点击按钮“确定”的返回值为 1。
        Form.Login
    End If
End Sub
```

```
End If
End Sub
```

相关函数

Logout

6.10 Logout

Logout 函数用于注销当前用户。

语法

```
Form.Logout
```

举例

在流程图中添加按钮 Button3，单击该按钮将弹出对话框“是否注销系统？”。单击对话框中的“确定”将弹出二次确认框，确定后注销系统，否则取消操作。

```
Sub Button3_OnLButtonUp(nFlag, x, y)
Dim needLogout
    needLogout = App.MsgBox ("是否注销系统? ", " SCADA ", 1)
    If needLogout=1 Then
        Form.Logout
    End If
End Sub
```

相关函数

Login

6.11 OpenPic

OpenPic 用于打开指定的流程图画面。

语法

```
Form.OpenPic(strPicName)
```

参数

strPicName 应为“画面名称.pic”形式的流程图画面名称。

举例

打开流程图画面 aaa.pic。
Form.OpenPic("aaa.pic")

注意

OpenPic 为打开流程图画面。ShowWindow 为弹出流程图画面。

弹出与打开画面的区别为操作人员可以对弹出画面进行移动等操作，而对于打开画面则不能进行移动等操作，且打开的画面一定是全屏显示。

相关函数

ClosePopPic、ShowWindow

6.12 PrintScreen

PrintScreen 函数用于打印当前屏幕的显示信息。

语法

Form.PrintScreen

举例

在流程图中添加按钮 ButtonPrintScreen，单击该按钮将打印屏幕。

```
Sub ButtonPrintScreen_OnLButtonUp(nFlag, x, y)
    Form.PrintScreen ' 打印屏幕
End Sub
```

6.13 Quality

Quality 函数用于获取指定位号的质量码。

语法

Form.Quality (tagName)

参数

tagName，位号名，采用字符串形式，支持结构位号。结构位号的格式为“位号名.位号的字段”，如“A1.Value”中“A1”为位号名、“Value”为字段。

返回值

0 表示指定位号正常，0x80000000 表示指定位号 Bad。

举例

获取位号 tag1 的质量码。

```
Form.Quality ("tag1")
```

相关函数

GetTagOffColor、GetTagOnColor、GetTagOffDesc、GetTagOnDesc

6.14 ShowLayer

ShowLayer 函数用于显示或隐藏指定的图层。

语法

```
Form.ShowLayer(nLayer,bShow)
```

参数

nLayer, 为图层编号, 有效值为 0~3。

bShow, 为图层的显示属性。False 表示隐藏图层, True 表示显示图层。

举例

显示图层 0 中的所有对象。

```
Form.ShowLayer 0,True
```

隐藏图层 0 中的所有对象。

```
Form.ShowLayer 0,False
```

6.15 ShowWindow

ShowWindow 函数用于弹出指定的流程图画面。

语法

```
Form.ShowWindow (strPicName)
```

参数

strPicName 应为“画面名称.pic”形式的流程图画面名称。

举例

弹出画面 aaa.pic。

```
Form.ShowWindow ("aaa.pic")
```

注意

OpenPic 为打开流程图画面。ShowWindow 为弹出流程图画面。

弹出与打开画面的区别为操作人员可以对弹出画面进行移动等操作, 而对于打开画面则不能进行移动等操作, 且打开的画面一定是全屏显示。

相关函数

OpenPic

6.16 ShutDown

ShutDown 函数用于退出监控系统。

语法

```
Form.ShutDown
```

举例

在流程图中添加按钮 ButtonShutdown，单击该按钮将退出监控系统。

```
Sub ButtonShutdown_OnLButtonUp(nFlag, x, y)
    Form.Shutdown      '退出系统
End Sub
```

6.17 Tag

Tag 函数用于对指定的单个位号进行读、写操作。

语法

```
Form.Tag(tagName)
```

参数

tagName，位号名，采用字符串形式，支持结构位号。结构位号的格式为“位号名.位号的字段”，如“A1.Value”中“A1”为位号名、“Value”为字段。

举例

将模拟量位号 A 的实时值更改为 12。

```
Form.Tag("A")=12
```

将开关量位号 D 的实时值更改为 True。

```
Form.Tag("D")=TRUE
```

定义变量 MyTag，并将自定义变量 T1 的值写入到 MyTag。

```
Dim MyTag
MyTag = Form.Tag("T1")
```

注意

对 IO 位号进行写操作时，写入值不能超量程。

6.18 TagDecimal

TagDecimal 函数用于获取指定位号的小数位数。

语法

```
Form.TagDecimal(tagName)
```

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“THIS.LIMIT”中“THIS”为位号名、“LIMIT”为字段。

返回值

返回值, 位号的小数位数。

若小数位缺省, 则显示的小数位数即为在系统结构组态软件的全局默认配置的小数位。默认为 3。

举例

在流程图中添加文本框 Text1 和按钮 Button2, 单击按钮将位号 MEM00001 的小数位数显示到文本框 Text1 中。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
Dim Decimal
    Decimal = Form.TagDecimal("MEM00001")
    Text1.Text = Decimal      ' 显示到文本框中
End Sub
```

6.19 TagDescription

TagDescription 函数用于获取指定位号的描述信息。

语法

Form.TagDescription (tagName)

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“THIS.LIMIT”中“THIS”为位号名、“LIMIT”为字段。

返回值

位号的描述。

举例

在流程图中添加文本框 Text1 和按钮 Button2, 单击 Button2 将位号 MEM00001 的描述信息显示到文本框 Text1 中。

当代码生效时, 点击按钮, 即可将位号的描述信息显示在文本对象中。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
    Text1.Text = Form.TagDescription("MEM00001")      ' 显示到文本框中
End Sub
```

6.20 TagN

TagN 函数用于对指定的单个位号进行读、写操作, 且操作记录不计入日志。

语法

```
Form.TagN(tagName)
```

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“A1.Value”中“A1”为位号名、“Value”为字段。

举例

将模拟量位号 A1 的实时值更改为 12, 且不将操作记录写入日志。

```
Form.TagN("A1")=12
```

注意

对 IO 位号进行写操作时, 写入值不能超量程。

相关函数

Tag

6.21 TagName

TagName 函数用于获取指定位号的实际位号名。

语法

```
Form.TagName(tagName)
```

参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“A1.Value”中“A1”为位号名、“Value”为字段。

举例

在流程图中添加按钮 Button2, 单击该按钮将获取@MYTAG@的实际位号名, 并对该位号写值 100。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
Dim tagName
    tagName = Form.TagName("@MYTAG@")      ' @MYTAG@在点组文件中定义
    App.WriteTag tagName, 100
End Sub
```

6.22 GoForward

GoForward 函数用于使监控画面前进一页, 只有经过后退 (GoBack) 的监控画面才能前进, 否则无效。

语法

```
Form.GoForward
```

举例

在流程图中添加按钮 11，单击按钮，则跳转到下一个监控画面（趋势画面或流程图画面）。

```
Sub Button11_OnLButtonUp(nFlag, x, y)
    Form.GoForward
End Sub
```

相关函数

GoBack

6.23 GoBack

GoBack 函数用于使监控画面后退一页。

语法

```
Form.GoBack
```

举例

在流程图中添加按钮 Button11，单击按钮，则后退到前一个监控画面（趋势画面或流程图画面）。

```
Sub Button11_OnLButtonUp(nFlag, x, y)
    Form.GoBack
End Sub
```

相关函数

GoForward

6.24 GotoFatherNodeHomePic

GotoFatherNodeHomePic 函数用于层级间的流程图跳转，实现打开上层第一页的功能。

语法

```
Form.GotoFatherNodeHomePic
```

举例

在流程图中添加按钮 Button1，单击按钮，则打开流程图所在层的上一层级的第一页，对应页不存在时不响应。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    Form.GotoFatherNodeHomePic
End Sub
```

相关函数

GotoChildNodeHomePic、GotoPrePic、GotoNextPic、GotoNodeHomePic、GotoPreNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.25 GotoChildNodeHomePic

GotoChildNodeHomePic 函数用于层级间的流程图跳转，实现打开下层第一页的功能。

语法

```
Form.GotoChildNodeHomePic
```

举例

在流程图中添加按钮 Button2，单击按钮，则打开流程图所在层的下一层级的第一页，对应页不存在时不响应。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
Form.GotoChildNodeHomePic
End Sub
```

相关函数

GotoFatherNodeHomePic、GotoPrePic、GotoNextPic、GotoNodeHomePic、GotoPreNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.26 GotoPrePic

GotoPrePic 函数用于层级间的流程图跳转，实现打开同层上一页的功能。

语法

```
Form.GotoPrePic()
```

举例

在流程图中添加按钮 Button3，单击按钮，则打开流程图所在层的上一页流程图，对应页不存在时不响应。

```
Sub Button3_OnLButtonUp(nFlag, x, y)
Form.GotoPrePic()
End Sub
```

相关函数

GotoFatherNodeHomePic 、 GotoChildNodeHomePic 、 GotoNextPic 、 GotoNodeHomePic 、 GotoPreNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.27 GotoNextPic

GotoNextPic 函数用于层级间的流程图跳转，实现打开同层下一页的功能。

语法

```
Form.GotoNextPic()
```

举例

在流程图中添加按钮 Button4，单击按钮，则打开流程图所在层的下一页流程图，对应页不存在时不响应。

```
Sub Button4_OnLButtonUp(nFlag, x, y)
Form.GotoNextPic()
End Sub
```

相关函数

GotoFatherNodeHomePic 、 GotoChildNodeHomePic 、 GotoPrePic 、 GotoNodeHomePic 、 GotoPreNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.28 GotoNodeHomePic

GotoNodeHomePic 函数用于层级间的流程图跳转，实现打开某层级第一页的功能。

语法

```
Form.GotoNodeHomePic(nodeName)
```

举例

在流程图中添加按钮 Button5，单击按钮，则打开层级名为“流程图小组 11”下的第一页，对应页不存在时不响应。

```
Sub Button5_OnLButtonUp(nFlag, x, y)
Form.GotoNodeHomePic("流程图小组 11")
End Sub
```

相关函数

GotoFatherNodeHomePic 、 GotoChildNodeHomePic 、 GotoPrePic 、 GotoNextPic 、 GotoPreNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.29 GotoPreNodeHomePic

GotoPreNodeHomePic 函数用于层级间的流程图跳转，实现打开前一个相邻层第一页的功能。

语法

```
Form.GotoPreNodeHomePic
```

举例

在流程图中添加按钮 **Button6**，单击按钮则打开流程图所在层的前一个相邻层级的第一页，对应页不存在时不响应。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Form.GotoPreNodeHomePic
End Sub
```

相关函数

GotoFatherNodeHomePic、GotoChildNodeHomePic、GotoPrePic、GotoNextPic、GotoNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.30 GotoNextNodeHomePic

GotoNextNodeHomePic 函数用于层级间的流程图跳转，实现打开后一个相邻层第一页的功能。

语法

```
Form.GotoNextNodeHomePic
```

举例

在流程图中添加按钮 **Button7**，单击按钮，则打开流程图所在层的后一个相邻层级的第一页，对应页不存在时不响应。

```
Sub Button7_OnLButtonUp(nFlag, x, y)
Form.GotoNextNodeHomePic
End Sub
```

相关函数

GotoFatherNodeHomePic、GotoChildNodeHomePic、GotoPrePic、GotoNextPic、GotoNodeHomePic、GotoPreNodeHomePic、GotoPreNodeThisPic、GotoNextNodeThisPic

6.31 GotoPreNodeThisPic

GotoPreNodeThisPic 函数用于层级间的流程图跳转，实现跳转到所在层上一层向上相邻层相同位置的特征页的功能。

语法

```
Form.GotoPreNodeThisPic("分隔符")
```

参数

分隔符可自定义。

特征页：分隔符前名称不同，分隔符后的名称相同，比如层级 1-流程图 1 和层级 2-流程图 1 即为特征页。

举例

组态如下：

- 在流程图下创建两个两层以上的分组，如下图所示。



图 6-2 创建流程图分组

- 在第二层“流程图小组 11”中创建流程图名为“流程图小组 11-1”，并在流程图中添加 2 个按钮 Button8 和 Button9，并增加以下脚本。
- 在“流程图小组 21”中创建流程图名为“流程图小组 21-1”，并在流程图中添加 2 个按钮 Button8 和 Button9，并增加以下脚本。

监控运行时，点击“流程图小组 11-1”中的按钮 8 无响应（因为向上无层级），点击按钮 9 则将跳转到“流程图小组 21-1”；点击“流程图小组 12-1”中的按钮 8 将跳转到“流程图小组 11-1”，点击按钮 9 无响应（因为向下无层级）。

```
Sub Button8_OnLButtonUp(nFlag, x, y)
    Form.GotoPreNodeThisPic("-")
End Sub
Sub Button9_OnLButtonUp(nFlag, x, y)
    Form.GotoNextNodeThisPic("-")
End Sub
```

相关函数

GotoFatherNodeHomePic、GotoChildNodeHomePic、GotoPrePic、GotoNextPic、GotoNodeHomePic、GotoPreNodeHomePic、GotoNextNodeHomePic、GotoNextNodeThisPic

6.32 GotoNextNodeThisPic

GotoNextNodeThisPic 函数用于层级间的流程图跳转，实现跳转到所在层上一层向下相邻层相同位置的特征页的功能。

语法

```
Form.GotoNextNodeThisPic("分隔符")
```

举例

请参看 GotoPreNodeThisPic。

相关函数

GotoFatherNodeHomePic、GotoChildNodeHomePic、GotoPrePic、GotoNextPic、GotoNodeHomePic、GotoPreNodeHomePic、GotoNextNodeHomePic、GotoPreNodeThisPic

7 APP 对象函数

App 对象生命周期和监控系统相同，不会在翻页后消失，可以用于页面间传递参数和获取一些系统参数。

App 对象函数语法为：App.函数名称

7.1 Alarm

Alarm 函数用于获取实时报警和历史报警数据。

语法

App.Alarm

参数

无

返回值

报警的集合形式，包含 RTAlarm 参数和 HistoryAlarm 参数。

RTAlarm 参数为报警的实时信息，支持 SetRTAlmFilter 或 SetRTAlmFilterEx 函数设置过滤条件，支持 GetRTAlmCount 函数获取过滤信息。

HistoryAlarm 参数为报警的历史信息，支持 QueryAlarm 函数对其进行查询。

举例 1--历史报警查询

代码：

```
Set a = App.Alarm
```

```
Set b = a.HistoryAlarm
```

```
Set c = b.QueryAlarm ("2014-12-5 09:30:00", "2014-12-5 09:40:00", 100, 0)
```

```
c.MoveFirst
```

```
Do Until c.EOF
```

```
    App.MsgBox "报警位号： "& c.tagName & chr(10) & "报警时间： " & c.Time & chr(10) & "报警状态： " & c.Status, "Value", 0 '两个内容之间加 “chr(10)” 表示分行显示
```

```
    c.MoveNext
```

```
Loop
```

说明：

QueryAlarm(beginTime, endTime, maxCount, timeout) 中 4 个参数含义如下：

- beginTime： 开始时间
- endTime： 终止时间
- maxCount： 最大查询个数
- timeout： 超时， 单位： 毫秒

查询从开始时间到终止时间内的历史报警记录，最大返回 100 条记录，弹出显示获取的位号名。

例子中 c 是报警的集合，通过 c 的 MoveFirst 和 MoveNext 可以遍历集合，Eof 判断是否读到集合尾部。

c 的成员变量（即为历史报警位号信息）如下：

- TagName: 位号名
- Time: 报警时间
- Description: 报警描述
- Status: 报警状态
- Serverity: 报警优先级
- AckTime: 报警确认时间
- VanishTime: 报警消除时间
- Zone: 报警分区

上述代码生效后，在监控见面中显示如下对话框，点击“确定”依次显示下一条信息直到完成。



图 7-1 历史报警查询

举例 2--实时报警统计 1

代码：

```
Set a = App.Alarm
Set b = a.RTAlarm
b.SetRTAlmFilter "PRI(0,31)"
Text1.Text = b.GetRTAlmCount
```

说明：

先使用 SetRTAlmFilter 接口设置查询条件，例子中设置的查询条件为查询报警优先级在 0~31 中的报警信息，如果需要过滤多个条件，则可用&连接，如 b.SetRTAlmFilter "PRI(0,31)&ACK(0)"，条件字符串可参看表 13-8 报警统计 RTAC 条件字符串。

再使用 GetRTAlmCount 获得实时报警总数。

举例 3--实时报警统计 2

代码：

```
Set a = App.Alarm
Set b = a.RTAlarm
b.SetRTAlmFilterEx("RTALMSTAT('PRI(0,10)').Max('AlmTime')")
result = b.GetRTStatResult()
app.MsgBox CStr(result), "MaxAlmTime", 0
```

说明：

先使用 SetRTAlmFilter 接口设置查询条件，例子中设置的查询条件为查询报警优先级在 0~10 中报警确认时间最晚的报警信息，并将报警信息通过提示框的形式显示，如果需要过滤多个条件，则可用&连接，如 b.SetRTAlmFilterEx("RTALMSTAT("PRI(0,10)&ACK(0)").Max("AlmTime"))，条件和子条件字符串可参看

表 13-9 报警统计 RTAlmStat 条件字符串，表 13-10 报警统计 RTAlmStat 返回值的属性和方法。再使用 GetRTAlmCount 获得实时报警总数。

注意：SetRTAlmFilterEx 兼容 SetRTAlmFilter 的功能，当 SetRTAlmFilterEx 替代 SetRTAlmFilter 使用时，代码应写成如下所示：

```
set a = app.Alarm
set b = a.RTAlarm
b.SetRTAlmFilterEx("RTAC("PRI(0,31)"))
MSGBOX b.GetRTStatResult
```

举例 4--单条报警确认

代码：

```
Sub Button1_OnLButtonDown(nFlag, x, y)
    set a = App.Alarm
    set b = a.RTAlarm
    b.Acknowledge "B", 1024 '位号 B 的高高限报警确认
End Sub

Sub Button2_OnLButtonDown(nFlag, x, y)
    set a = App.Alarm
    set b = a.RTAlarm
    b.AcknowledgeByType "B", "LL" '位号 B 的低低限报警确认
End Sub
```

说明：

在流程图中加两个按钮对象（Button1~Button2）。。

上述语句生效后，点击按钮 1 确认位号 B 的高高限报警，按钮 2 则确认其低低限报警。

Acknowledge (TagName, dwSubCondition) / AcknowledgeByType (TagName, almType)：根据子条件（dwSubCondition）或报警类型（almType）确认某位号（TagName）的某条报警。各类报警对应的子条件如表 13-11 所示。

相关参数

GetRTAlmCount, SetRTAlmFilter, QueryAlarm

7.2 Database

Database 函数用于获取位号的历史数据。

语法

App.Database

参数

无

返回值

历史数据的集合形式，支持使用 QueryHistoryData 函数对返回值进行查询。

举例

代码：

```
Set a = App.Database
Set b = a.HistoryData
Set c = b.QueryHistoryData("TAG0", "2014-7-28 08:00:00", "2014-7-28 09:00:00", 1, 0)
c.MoveFirst
Do Until c.EOF
    App.MsgBox c.tagName, "Value", 0
    c.MoveNext
Loop
```

说明：

历史趋势查询方法和查询历史报警类似。

QueryHistoryData(tagName, beginTime, endTime, interval, timeout)中的参数含义如下：

- tagName: 位号名
- beginTime: 开始时间
- endTime: 终止时间
- interval: 时间间隔，范围（1~60）s
- timeout: 超时，单位：毫秒

例子中 c 是报警的集合，通过 c 的 MoveFirst 和 MoveNext 可以遍历集合，Eof 判断是否读到集合尾部。

然后遍历 c 集合，c 的成员变量如下：

- tagName: 位号名
- Quality: 质量码
- tagValue: 值
- Time: 时间

相关函数

QueryHistoryData

7.3 Exit

Exit 函数用于退出指定用户在监控中的登录。

语法

```
App.Exit(userName,userPassword)
```

参数

userName，用户名（字符串）。

userPassword，用户密码（字符串）。

需要注意的是，当无输入参数时，将在界面中显示所有的用户名。根据显示的用户名，选择用户名并输入密码后退出。

返回值

整数，1 表示退出成功，-1 表示密码错误，-2 表示用户不存在，-3 表示无退出权限。

举例

在流程图中添加按钮 Button6，单击按钮 Button6 后 admin 用户将退出监控系统。若 admin 用户的用户名或密码错误将弹出提示框“用户名、密码错误或用户无退出权限”。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Dim ret
ret = App.Exit("admin","admin") '返回值：1 成功，-1 密码错误，-2 用户不存在，-3 没有退出权限
If ret < 1 Then '若用户名和密码错误，则弹出信息框
    App.MsgBox "用户名、密码错误或用户无退出权限", "SCADA", 0
End If
End Sub
```

7.4 GetCurrentUser

GetCurrentUser 函数用于获取当前监控系统的登录用户信息。

语法

```
App.GetCurrentUser
```

参数

无

返回值

字符串形式的登录用户名。

举例

在流程图中添加按钮 Button6，单击该按钮将弹出提示框显示当前的登录用户没有权限，请切换至管理员用户。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Dim user
user = App.GetCurrentUser
```

```
App.MsgBox "用户" & user & "没有权限，请切换至管理员用户", "SCADA", 0  
End Sub
```

注：语句可用于有权限控制的操作中，配合获取用户权限接口函数使用。

7.5 GetCurrentUserGrade

GetCurrentUserGrade 函数用于获取当前监控系统的登录用户等级。

语法

```
App.GetCurrentUserGrade
```

参数

无

返回值

长整型的监控用户等级。

举例

在流程图中添加按钮 Button6，当登录用户的权限等级小于 5 时，单击该按钮将弹出提示框显示当前的登录用户权限不足。

```
Sub Button6_OnLButtonUp(nFlag, x, y)  
Dim grade  
grade = App.GetCurrentUserGrade  
If grade < 5 Then  
    App.MsgBox "权限不足", "SCADA", 0  
End If  
End Sub
```

7.6 GetGroup

GetGroup 函数用于获取当前监控登录的操作小组。

语法

```
App.GetGroup
```

参数

无

返回值

字符串形式的操作小组。

举例

在流程图中添加按钮 Button6，单击该按钮将弹出提示框显示当前登录的操作小组名。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Dim group
group = App.GetGroup
App.MsgBox "当前操作小组是" & group, "SCADA", 0
End Sub
```

7.7 GetServerState

GetServerState 函数用于获取服务器主从状态值。

语法

```
App.GetServerState(nodeIP)
```

参数

nodeIP, 计算机 IP（字符串），为空时表示获取本机服务器是主是从。

返回值

0 未知状态, 1 主服务器, 2 从服务器。

举例

在流程图添加按钮（Button6），单击按钮，将出现提示本机是否是主服务器。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Dim value
Value = App.GetServerState("")
If value = 1 Then
    App.MsgBox "本机是主服务器", "SCADA", 0
Elseif value = 2 Then
    App.MsgBox "本机不是主服务器", "SCADA", 0
End If
End Sub
```

7.8 GetTagLimitHigh

GetTagLimitHigh 函数用于获取指定位号的工程值上限。

语法

```
App.GetTagLimitHigh(name)
```

参数

name, 字符串形式的位号名。

返回值

位号工程值上限，双整型。

举例

在流程图中添加按钮 Button6，单击该按钮文本框 Text1 中获取位号 tag1 的工程值上限，并将其写入文本框 Text1 中。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
    Text1.text=App.GetTagLimitHigh("tag1")
End Sub
```

7.9 GetTagLimitLower

GetTagLimitLower 函数用于获取指定位号的工程值下限。

语法

```
App.GetTagLimitLower(name)
```

参数

name，字符串形式的位号名。

返回值

位号工程值下限，双整型。

举例

在流程图中添加按钮 Button6，单击该按钮文本框 Text1 中获取位号 tag1 的工程值下限，并将其写入文本框 Text1 中。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
    Text1.text=App.GetTagLimitLower("tag1")
End Sub
```

7.10 GetTagUnitByName

GetTagUnitByName 函数用于获取指定位号的单位。

语法

```
App.GetTagUnitByName(name)
```

参数

name，字符串形式的位号名。

返回值

字符串形式的位号单位。

举例

在流程图中添加按钮 Button6，单击该按钮文本框 Text1 中获取位号 tag1 的值及单位，并将其写入文本框 Text1 中。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Text1.text= App.ReadTag ("tag1")&App.GetTagUnitByName ("tag1")
End Sub
```

执行该函数后，若 Text1 中显示 70%则表示 tag1 的值为 70，单位为%。

7.11 GetTickCount

GetTickCount 函数用于获得系统已运行的时间（单位：ms）。

语法

```
App.GetTickCount
```

返回值

系统运行的时间，整型。

举例

在流程图中添加按钮 Button2，单击按钮，将弹出标题栏为“SCADA”的消息框，提示“系统已运行了 XXX 毫秒”。

```
Sub Button2_OnLButtonDown(nFlag, x, y)
tc = App.GetTickCount
App.MsgBox "系统已运行了" & CStr(tc) & "毫秒", "SCADA", 0
End Sub
```

7.12 GetUser

GetUser 函数用于获取指定用户索引值的用户名。

语法

```
App.GetUser(userIndex)
```

参数

userIndex，用户索引值，从 0 开始的整型数。

返回值

用户名，变体型。

举例

在流程图中添加按钮 **Button6**，单击该按钮文本框 **Text1** 中获取用户索引值为 0 的用户名，并将其写入文本框 **Text1** 中。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Text1.text=App.GetUser(0)
End Sub
```

7.13 GetUserCount

GetUser 函数用于获取当前登录监控的用户个数。

语法

```
App.GetUserCount
```

参数

无

返回值

用户个数，变体型。

举例

在流程图中添加按钮 **Button8**，单击该按钮将弹出提示框显示当前所有用户，用户间用,隔开。

```
Sub Button8_OnLButtonUp(nFlag, x, y)
Dim userCount
userCount = App.GetUserCount ' 获取用户个数。
Dim maxIndex
maxIndex = userCount - 1 '获取的用户个数减 1。
Dim users
For i=0 To maxIndex '从 0 开始到
    If users<>"" Then 'users 不为空时。
        users = users & "," '用户间用逗号隔开。
    End If
    users = users + App.GetUser(i)
Next
App.MsgBox "存在以下用户： " & users, "SCADA", 0
End Sub
```

7.14 GetUserTeam

GetUserTeam 函数用于获取指定用户所在的操作小组名称。

语法

```
App.GetUserTeam(userName,teamIndex)
```

参数

userName, 用户名（字符串）。

teamIndex, 操作小组索引值（长整型），从 0 开始。

返回值

操作小组名称，变体型。

举例

在流程图中添加按钮 Button8，单击该按钮后弹出提示框显示 admin 用户可操作的小组名。当 admin 用户可操作多个小组时，小组名间用逗号隔开。

```

Sub Button8_OnLButtonUp(nFlag, x, y)
Dim Count
Count = App.GetUserTeamCount("admin")      '获取用户 admin 有操作权限的小组个数。
Dim maxIndex
maxIndex = Count - 1                        '获取的个数减 1。
Dim Teams
For i=0 To maxIndex                          '从 0 开始到遍历
    If Teams <> "" Then
        Teams = Teams & ","                  '小组间用逗号隔开。
    End If
    Teams = Teams + App.GetUserTeam("admin",i)
Next
App.MsgBox "用户 admin 具有操作权限的小组有: " & Teams, "SCADA", 0
End Sub

```

7.15 GetUserTeamCount

GetUserTeamCount 函数用于获取指定用户具有权限的操作小组个数。

语法

App.GetUserTeamCount(userName)

参数

userName, 字符串形式的用户名。

返回值

变体型，操作小组个数。

举例

详见 GetUserTeam。

7.16 IsPrimaryServer

IsPrimaryServer 函数用于判断本机是否为主服务器。

语法

App.IsPrimaryServer

参数

无

返回值

BOOL 型，true 表示本机为主服务器，false 表示本机非主服务器。

举例

如果本机是主服务器，则位号 A 置值 100。

```
If App.IsPrimaryServer<>False Then
    App.WriteTag "A", 100
End If
```

7.17 Log

Log 函数用于查询日志信息。

语法

App.Log

参数

无

返回值

日志信息的集合。

举例

代码：

```
Set a = App.Log
Set b = a.QueryLog ("2014-7-28 08:00:00", "2014-7-28 09:00:00", 0)
b.MoveFirst
Do Until b.EOF
    App.MsgBox b.Creator, "Value", 0
    b.MoveNext
Loop
```

说明：

遍历方法同报警和趋势。

QueryLog(beginTime, endTime, timeout)中的参数如下：

- beginTime: 开始时间
- endTime: 终止时间
- timeout: 超时，单位：毫秒

例子中 b 是报警的集合，通过 b 的 MoveFirst 和 MoveNext 可以遍历集合，Eof 判断是否读到集合尾部。

遍历 b 集合，b 的成员变量如下：

- Creator: 对象
- event: 事件内容
- Host: 主机名
- Time: 时间
- type: 类型
- User: 用户

7.18 Login

Login 函数用于执行指定用户登录到指定操作小组的操作。

语法

```
App.Login(userName,userPassword,userTeam)
```

参数

userName, 用户名，字符串形式。

userPassword, 用户密码，字符串形式。

userTeam, 操作小组名，字符串形式。

返回值

BOOL 型，1 表示用户登录操作小组成功，0 表示用户登录操作小组失败。

举例

在流程图中添加按钮 Button2，单击该按钮用户 admin 将登录到操作小组 Team0001。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
App.Login "admin", "admin", "Team0001"
End Sub
```

7.19 Logout

Logout 函数用于注销当前登录的用户。

语法

App.Logout

参数

无

返回值

1 为注销成功，0 为注销失败。

举例

在流程图中添加按钮 Button2，单击该按钮将弹出提示框“是否注销系统”。单击提示框中的“确定”，将退出系统。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
Dim needLogout
needLogout = App.MsgBox("是否注销系统? ", "SCADA", 1)
If needLogout=1 Then          'msgbox 中点击“确定”按钮返回值为 1
    App.Logout
End If
End Sub
```

7.20 MsgBox

MsgBox 函数用于弹出消息框。

语法

```
App.MsgBox(prompt,title,buttons)
```

参数

- prompt，消息框中的提示内容，字符串形式。
- title，消息框标题栏，字符串形式。
- buttons，消息框按钮样式。消息框按钮样式，Button 值与样式的关系请参见“Msgbox 中按钮设置和返回值说明”。

返回值

长整型，返回值与按钮的对应关系如下表所示。

值	按钮
1	确定
2	取消
3	放弃
4	重试
5	忽略

值	按钮
6	是
7	否

举例

在流程图中添加按钮 Button1，单击该按钮将弹出标题为“提示框”，显示内容为“即将弹出流程图 1”的弹出框。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
App.MsgBox "即将弹出流程图 1", "提示框", 0
End Sub
```

在监控中，单击按钮 Button1 将弹出如下图所示的提示框。



注意事项

在流程图和弹出式流程图中不允许直接使用 MsgBox 函数，请使用 App.MsgBox。

7.21 Mute

Mute 函数用于进行报警消音。

语法

```
App.Mute
```

参数

无

返回值

无

举例

在流程图中添加按钮 ButtonMute，单击该按钮将进行报警消音。

```
Sub ButtonMute_OnLButtonUp(nFlag, x, y)
App.Mute ' 报警消音
End Sub
```

7.22 OpenPic

OpenPic 函数用于打开指定的流程图。

语法

```
App.OpenPic(fullPictureName)
```

参数

fullPictureName，流程图名，字符串形式。

返回值

无

举例

在流程图中添加按钮 Button1、Button2，单击 Button1 将弹出流程图“area1”，单击 Button2 将弹出流程图“area2”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    App.OpenPic("area1.pic")
End Sub
Sub Button2_OnLButtonUp(nFlag, x, y)
    App.OpenPic "area2.pic"
End Sub
```

7.23 PopupPic

PopupPic 函数用于打开指定的弹出式流程图。

语法

```
App.PopupPic(PictureName)
```

参数

PictureName，弹出式流程图名，字符串形式。

返回值

无

举例

在流程图中添加按钮 Button1，单击该按钮将弹出弹出式流程图“area2”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    App.PopupPic("area2.pic")
End Sub
```

7.24 PopupPicEx

PopupPicEx 函数用于打开指定的弹出式流程图。

语法

```
App.PopupPicEx(PictureName, xPos, yPos)
```

参数

PictureName, 弹出式流程图名, 字符串形式, xPos, yPos 坐标信息。

返回值

无

举例

在流程图中添加按钮 Button1, 单击该按钮将弹出弹出式流程图 “area2”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    App.PopupPicEx "area2.pic", 100, 200
End Sub
```

7.25 PrintScreen

PrintScreen 函数用于打印当前屏幕信息。

语法

```
App.PrintScreen
```

参数

无

返回值

无

举例

在流程图中添加按钮 Button1, 单击该按钮打印当前屏幕信息。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    App.PrintScreen
End Sub
```

7.26 ReadTag

ReadTag 函数用于读取指定位号的值。

语法

`App.ReadTag(tagName)`

输入参数

`tagName`, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“THIS.LIMIT”中“THIS”为位号名、“LIMIT”为字段。

输出参数

变体型, 位号的值。

返回值

无

注意

本函数不推荐使用, 为系统保留函数。

当需要读取位号的值时, 可以使用函数 `ReadTagEx`, 详情参见“7.27 `ReadTagEx`”。

7.27 ReadTagEx

`ReadTagEx` 函数用于同步读取位号值, 若同步读取失败则延时指定时间读取。

语法

`App.ReadTagEx(tagName,timeout)`

输入参数

`tagName`, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“THIS.LIMIT”中“THIS”为位号名、“LIMIT”为字段。

`timeout`, 长整型, 读取位号的延时时间, 单位为 ms。

输出参数

变体型, 位号的值。

返回值

无

举例

在流程图中添加文本框 `Text1`, 使用 `ReadTagEx` 函数读取位号的 `MEM00001` 的值。若位号有值则返回位号值, 若位号无值则等待 3 秒读取。

```
Dim tagValue
```

```
tagValue = App.ReadTagEx("MEM00001", 3000)
```

```
Text1.Text = tagValue
```

注意

InPlant SCADA 中，App.ReadTagAsy("MEM00001")效果等同于 App.ReadTagEx("MEM00001", 3000)。超时时间为 3s。

相关函数

ReadTag, ReadTagAsy

7.28 ReadTagAsy

ReadTagAsy 函数用于同步读取位号的值。

语法

```
App.ReadTagAsy(tagName)
```

输入参数

tagName, 位号名, 采用字符串形式, 支持结构位号。结构位号的格式为“位号名.位号的字段”, 如“THIS.LIMIT”中“THIS”为位号名、“LIMIT”为字段。

输出参数

变体型, 位号的值。

返回值

无

举例

在流程图中添加文本框 Text1, 使用 ReadTagAsy 函数同步读取位号的 MEM00001 的值。

```
Dim tagValue  
tagValue = App.ReadTagAsy("MEM00001")  
Text1.Text = tagValue
```

相关函数

ReadTag, ReadTagEx

7.29 ReadTags

ReadTags 函数用于批量查询位号值。

语法

```
App.ReadTags(vTags,pvValue,pvQuality)
```

输入参数

vTags: 位号名（变体型）。
pvValue: 位号值（变体型）。
pvQuality: 位号质量码。

输出参数

变体型，位号的值。

返回值

BOOL 型，true 表示读取位号成功，false 表示读取位号失败。

举例

在流程图中添加按钮 **Button6**，单击该按钮将出现提示框显示两个位号值的和。

```
Sub Button6_OnLButtonUp(nFlag, x, y)
Dim tags(1)
Dim values(1)
Dim qualities(1)
tags(0) = "MEM00001" '定义位号名 1
tags(1) = "MEM00002" '定义位号名 2
ret = App.ReadTags (tags,values,qualities) '批量获取上述两个位号值
If ret = True Then
    For Each i In values
        App.MsgBox cstr(i), "SCADA", 0
    Next
End if
If ret = False then
    App.MsgBox "获取位号值失败！ ", "SCADA", 0
End If
End Sub
```

7.30 ReadValue

ReadValue 函数用于读取全局变量的值。

语法

App.ReadValue (name)

输入参数

name，全局变量名，采用字符串形式。

返回值

变体型，读取变量成功则返回变量被赋予的值，读取变量失败则返回 `empty`。

举例

从全局变量 `MyValue` 中读取数值，如果读取成功 `b = 100`，否则返回 `empty`。

```
b = App.ReadValue("MyValue")
```

7.31 SaveValue

`SaveValue` 函数用于将指定值写入到全局变量。

语法

```
App.SaveValue (name,value)
```

输入参数

`name`，全局变量名，采用字符串形式。

`value`，要写入到变量的值。

举例

将 100 保存到全局变量 `MyValue`。

```
App.SaveValue "MyValue",100
```

7.32 ServerInfo

`ServerInfo` 函数用于获取服务器 IP。

返回值

服务器 IP，字符串。

举例

在流程图中加一个按钮对象（`Button1`）和三个文本对象（`Text1`、`Text2`、`Text3`）。单击按钮，则在文本对象中分别显示主从服务器 IP 地址，若网络冗余则各显示两个 IP 地址，如图 7-2 所示。

```
主: 172.30.10.59 172.30.10.166
从: 172.20.1.195 172.20.1.200
当前: 172.30.10.166 172.30.10.59
```

图 7-2 获取主从服务器 IP

```
Sub Button1_OnLButtonDown(nFlag, x, y)
```

```
    ip1 = ""
```

```
    ip2 = ""
```

```
    ip3 = ""
```

```

Set a = App.ServerInfo
Set b = a.GetPrimaryServerIP '查找主服务器 IP
Set c = a.GetBackupServerIP '查找从服务器 IP
Set d = a.GetCurrentIP '查找当前计算机 IP
b.MoveFirst '循环语句，若主服务器网络冗余，则将显示两个 IP
do Until b.Eof
    ip1 = ip1 + " " + b.IPValue
    b.MoveNext
loop
Text1.Text = "主:"+ip1

c.MoveFirst '循环语句，若从服务器网络冗余，则将显示两个 IP
do Until c.Eof
    ip2 = ip2 + " " + c.IPValue
    c.MoveNext
loop
Text2.Text = "从:"+ip2

d.MoveFirst '循环语句，若当前计算机网络冗余，则将显示两个 IP
do Until d.Eof
    ip3 = ip3 + " " + d.IPValue
    d.MoveNext
loop
Text3.Text = "当前:"+ip3
End Sub

```

7.33 SetPrimaryServerIP

SetPrimaryServerIP 函数用于切换主从服务器。

语法

```
App.SetPrimaryServerIP(nodeIP)
```

输入参数

nodeIP，切换后的 IP，字符串形式。

若切换后的服务器为网络冗余，则两个冗余 IP 间可用“;”隔离，请参照“举例”。

返回值

1，表示切换成功；0 表示切换失败。

举例

在流程图添加以下脚本后，单击流程图中按钮 Button2 将主服务器切换为 127.0.0.1 或

127.0.0.2。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
    App.SetPrimaryServerIP "127.0.0.1;127.0.0.2" '两个 IP 属于同一个网络冗余的服务器
End Sub
```

7.34 WriteLog

WriteLog 函数用于写日志。

语法

```
App.WriteLog(type,object,event)
```

输入参数

type, 操作日志的类型, 字符串形式。

object, 目标名称, 字符串形式。

event, 事件名称, 字符串形式。

举例

在流程图中添加以下脚本后, 单击流程图中的按钮 **Button1** 将写入一条日志。日志的类型为“用户”、日志的对象为“用户登录”, 日志的事件为“用户 1 登录”。

```
Sub Button1_OnLButtonDown(nFlag, x, y)
    App.WriteLog"用户","用户登录","用户 1 登录"
End Sub
```

7.35 WriteTag

WriteTag 函数用于对位号写值, 写值动作不记录到日志。

语法

```
App.WriteTag tagName>tagValue
```

输入参数

tagName, 位号名, 字符串形式。

tagValue, 位号值, 变体型。

返回值

无

举例

对指定位号写入特定位号值, 此更改不记录在操作日志中; "Webreport_His_AO.OUT"为指定结构位号的位号名, "50"为指定写入的位号值。

```
App.WriteTag "Webreport_His_AO.OUT",50
```

7.36 WriteTagEx

WriteTagEx 函数用于对超时的位号写值，且写值动作记录到日志。

语法

```
App.WriteTagEx(tagName,timeout,tagValue)
```

输入参数

tagName，位号名，字符串形式。

timeout，写位号的超时时间，长整型。

tagValue，位号值，变体型。

举例

对位号 MEM00001 进行写值，此操作将记录在操作日志中。

```
App.WriteTagEx "MEM00001", 0, 100
```

7.37 WriteTagL

WriteTagL 函数用于对位号写值，且写值动作记录到日志。

语法

```
App.WriteTagL lpTagName, pvValue
```

输入参数

lpTagName，位号名，字符串形式。

pvValue，位号值，变体型。

注意

本函数不推荐使用，为系统保留函数。

当需要写位号的值时，可以使用函数 WriteTagEx，详情参见“7.36 WriteTagEx”。

8 图形对象函数

指 VFDraw 中工具菜单下的“图形对象浏览器”中的所有对象。

通过调用图形对象提供的脚本接口能够实现更强大的功能。图形对象的接口分为“公共接口”、“常用接口”与“专用接口”三部份。公共接口，指图形对象和控件都具有的接口；常用接口，指大多数图形对象所具有的、常用到的，但并不是所有图形对象都支持的接口；专用接口，只有某图形对象或控件才具有的接口。

当设置的图形对象边界超出屏幕时，在当前屏幕上将不可见。

图形对象函数语法为：图形对象名.接口名称。

8.1 公共接口

图形对象的公共接口，是指图形对象和控件都具有的接口。

另外，图形对象还包含常用接口和专用接口。关于常用接口的详细内容请参见“常用接口”，关于专用接口的详细内容请参见具体的控件说明章节。

8.1.1 Bottom

Bottom 函数用于设置图形对象的下边界。

语法

图形对象名.Bottom=下边界值

其中下边界值的有效范围为-4096 到 4096，单位为像素。

举例

将文本框的下边界定到屏幕上 20 像素的位置。

```
Text1.Bottom = 20
```

8.1.2 Height

Height 函数用于设置图形对象的高度。

语法

图形对象名.Height=高度值

其中高度值的有效范围为-4096 到 4096，单位为像素。当高度值为负值时，图形对象的高度按高度值的绝对值显示。

举例

将文本框的高度设定为 20 像素。

```
Text1.Height=20
```

8.1.3 Layer

Layer 函数用于设置图形对象的所在图层。

语法

图形对象名.Layer=图层编号

其中图层编号的有效范围为 0 到 3。

举例

将文本框的设置放在第一图层。

```
Text1.Layer= 1
```

8.1.4 Left

Left 函数用于设置图形对象的左边界。

语法

图形对象名.Left=左边界值

其中左边界值的有效范围为-4096 到 4096，单位为像素。

举例

将文本框的左边界定到屏幕上 20 像素的位置。

```
Text1.Left= 20
```

8.1.5 Right

Right 函数用于设置图形对象的右边界。

语法

图形对象名.Right=右边界值

其中右边界值的有效范围为-4096 到 4096，单位为像素。

举例

将文本框的右边界定到屏幕上 20 像素的位置。

```
Text1.Right= 20
```

8.1.6 Top

Top 函数用于设置图形对象的上边界。

语法

图形对象名.Top=上边界值

其中上边界值的有效范围为-4096 到 4096，单位为像素。

举例

将文本框的上边界定到屏幕上 20 像素的位置。

```
Text1.Top=20
```

8.1.7 Visible

Visible 函数用于设置图形对象的可视性。

语法

图形对象名.Visible=True 或 False

举例

将文本框设定为可视。

```
Text1.Visible=True
```

8.1.8 Width

Width 函数用于设置图形对象的宽度。

语法

图形对象名.Width=宽度值

其中宽度值的有效范围为-4096 到 4096，单位为像素。当高度值为负值时，图形对象的高度按高度值的绝对值显示。

举例

将文本框的宽度设定为 20 像素。

```
Text1.Width=20
```

8.2 常用接口

图形对象指的是画面绘制软件 VFDraw 中“图形对象浏览器”中的所有对象。图形对象的常用接口，是指大多数图形对象所具有的、常用到的，但并不是所有图形对象都支持的接口。

另外，图形对象还包含公共接口和专用接口。关于公共接口的详细内容请参见“公共接口”，关于专用接口的详细内容请参见具体控件的说明章节。

8.2.1 Angle

Angle 函数用于控制图形对象按顺时针旋转指定的角度。

语法

图形对象名.Angle=旋转的角度

举例

功能说明：将圆角矩形对象 RoundRect1 顺时针旋转 30 角度

```
RoundRect1.Angle = 30
```

8.2.2 Align

Align 函数用于设定图形对象中文本的显示位置。

语法

图形对象名.Align=显示位置。

显示位置的取值范围为 1~3。其中，1 表示左对齐，2 表示居中，3 表示右对齐。

举例

在流程图中添加按钮 Button1，当鼠标从 Button1 中移出时，文字框 Text1 的内容将居中显示。

```
Sub Button1_OnMouseLeave()  
    Text1.Align = 2  
End Sub
```

8.2.3 BackgroundColor

BackgroundColor 函数用于设定图形对象的背景色。

语法

图形对象名.BackgroundColor=背景色的值。

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

将当前圆角矩形的背景色置为黑色。

```
RoundRect1.BackgroundColor = vbBlack
```

8.2.4 BackgroundStyle

BackgroundStyle 函数用于设定图形对象的背景风格。

语法

图形对象名.BackgroundStyle=背景风格值。

背景风格值与具体风格的对应关系请参见“背景风格对照表”。

举例

将当前圆角矩形的背景风格设定为实心填充。

```
RoundRect1.BackgroundStyle= 0
```

8.2.5 Caption

Caption 函数用于设定图形对象的标题。

语法

图形对象名.Caption=标题内容，其中标题内容为字符串形式。

举例

将选择框 CheckBox1 的标题设定为 A。

```
CheckBox1.Caption="A"
```

8.2.6 CenterX

CenterX 函数用于设定图形对象的圆心横坐标。

语法

图形对象名.CenterX=坐标值。

其中，坐标值的取值范围为-4096~4096，单位为像素。当坐标值为 0 时，图形对象的圆心横坐标为画面的左边沿。

举例

设置椭圆对象 Ellipse1 的圆心横坐标位置为画面左边沿向右 20 像素。

```
Ellipse1.CenterX = 20
```

8.2.7 CenterY

CenterY 函数用于设定图形对象的圆心纵坐标。

语法

图形对象名.CenterY=坐标值。

其中，坐标值的取值范围为-4096~4096，单位为像素。当坐标值为 0 时，图形对象的圆心纵坐标为画面的上沿。

举例

设置椭圆对象 Ellipse1 的圆心纵坐标位置为画面上边沿向下 20 像素。

```
Ellipse1.CenterY = 20
```

8.2.8 EdgeColor

EdgeColor 函数用于设定图形对象的边框颜色。

语法

图形对象名.EdgeColor=颜色值。

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

将圆角矩形 RoundRect1 的边框置为红色。

```
RoundRect1.EdgeColor = vbRed
```

8.2.9 EdgeStyle

EdgeStyle 函数用于设定图形对象的边框样式。

语法

图形对象名.EdgeStyle = 风格值。

风格值与具体风格的对应关系请参见“边框风格对照表”。

举例

将圆角矩形 RoundRect1 的边框为实线。

```
RoundRect1.EdgeStyle = 0
```

8.2.10 EdgeWidth

EdgeWidth 函数用于设置对象的边框宽度。

语法

图形对象名.EdgeWidth = 数值

数值的有效范围是[0,6]。

举例

将线条的宽度设为 2 个像素

```
Line1.EdgeWidth = 2
```

8.2.11 Enabled

Enabled 函数用于设定图形对象的有效性。

语法

图形对象名.Enabled=Ture 或 False

True 表示图形对象有效，False 表示图形对象无效。

举例

设置按钮 Button1 有效。

```
Button1.Enabled=true
```

8.2.12 Font

Font 函数用于设定图形对象中文字的字体。

语法

图形对象名.Font=字体描述

举例

设置文本框 Text1 的字体与 Text2 的相同。

Text1.Font = Text2.Font '其中 Text2 的字体为宋体

8.2.13 FontColor

FontColor 函数用于设定图形对象中文字的字体颜色。

语法

图形对象名.FontColor =颜色值

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设置文本框 Text1 的字体颜色为红色。

Text1.FontColor= vbRed

8.2.14 GradientColor

GradientColor 函数用于设定图形对象的渐变色颜色。

语法

图形对象名.GradientColor =颜色值

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设置圆角矩形为红色渐变色。

RoundRect1.GradientColor = vbRed

8.2.15 GradientStep

GradientStep 函数用于设定图形对象的渐变步长。

语法

图形对象名.GradientStep=步长值

步长就是对象颜色渐变的像素，步长值为 1~16 的整数。

步长值为 1 时，对象就以 1 像素渐变；步长越小渐变效果越柔和，但是刷新时占用的 CPU 越高。

举例

设置圆角矩形为渐变步长为 2。

```
RoundRect1.GradientStep= 2
```

8.2.16 GradientStyle

GradientStyle 函数用于设定图形对象的渐变风格。

语法

图形对象名.GradientStyle=渐变风格值

渐变风格值对应的具体风格请参见“渐变风格对照表”。

举例

设置圆角矩形为渐变风格为由左至右的风格渐变。

```
RoundRect1.GradientStyle = 1
```

8.2.17 IsAutoSize

IsAutoSize 函数用于设定图形对象是否自动缩放。

语法

图形对象名.IsAutoSize=True 或 False

True 表示设定图形对象按设定的大小自动缩放，False 表示设定图形对象不进行自动缩放。

图形对象的大小是根据其 Height 和 Width 属性来确定的。

举例

设定文本框 Text1 按设定的大小自动缩放。

```
Text1.IsAutoSize = TRUE
```

相关函数

Height, Width

8.2.18 IsTransparent

IsTransparent 函数用于设定图形对象是否透明。

语法

图形对象名.IsTransparent=True 或 False

True 表示设定图形对象为透明，False 表示不设定图形对象为透明。

举例

设定文本框 Text1 为透明。

Text1.IsTransparent= TRUE

8.2.19 IsSemiTransparent

IsSemiTransparent 函数用于设定图形对象是否半透明。

语法

图形对象名.IsSemiTransparent =True 或 False

True 表示设定图形对象为半透明，False 表示不设定图形对象为半透明。

举例

设定圆角矩形对象 RoundRect1 为半透明。

RoundRect1.IsSemiTransparent = True

8.2.20 LineColor

LineColor 函数用于设定图形对象的线条颜色。

语法

图形对象名.LineColor =颜色值

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设定线条 Line1 的线条颜色为红色。

Line1.LineColor = vbRed

8.2.21 LineStyle

LineStyle 函数用于设定图形对象的线条风格。

语法

图形对象名.LineStyle=风格值

线条风格值为 0~5，0 为实线、1 为虚线、2 为点型虚线、3 为点线相间的虚线、4 为点线相间的虚线（一个虚线两个点）、5 为隐藏线。

举例

设定线条 Line1 的线条为实线。

Line1.LineStyle= 0

8.2.22 LineWidth

LineWidth 函数用于设定图形对象的线条宽度。

语法

图形对象名.LineWidth=宽度值
宽度值为 1~6 的整数，单位为像素。

举例

设定线条 Line1 的线条宽度为 4。
Line1.LineWidth = 4

8.2.23 Picture

Picture 函数用于设置“按钮”和“图片”的图片内容。

语法

图形对象名.Picture=图片路径及名称
宽度值为 1~6 的整数，单位为像素。

举例

设置按钮 Button1 的图片为本机 E 盘的 A.jpg
Button1.Picture = "E:\A.jpg"

8.2.24 RotateCenterX/RotateCenterY

RotateCenterX 函数用于设定图形对象旋转的圆心横坐标。
RotateCenterY 函数用于设定图形对象旋转的圆心纵坐标。

语法

图形对象名.RotateCenterX=横坐标值
图形对象名.RotateCenterY=纵坐标值

举例

圆角矩形以横坐标为 20 像素、纵坐标为 50 像素的某个点为圆心旋转。
RoundRect1.RotateCenterX = 20
RoundRect1.RotateCenterY = 50

8.2.25 SetTitle

SetTitle 函数用于设置面板或按钮的标题栏内容。

语法

图形对象名.SetTitle=标题内容
标题内容应设置在双引号""内。

举例

设置面板 Panel1 的标题为 A。

```
Panel1.SetTitle = "A"
```

8.2.26 SetTagValue

SetTagValue 函数用于设置图形对象关联位号的值。

语法

图形对象名.SetTagValue(位号值)

举例

设置多选框 CheckBox1 关联的位号值为 0。

```
CheckBox1.SetTagValue(0)
```

8.2.27 TagName

TagName 函数用于设置图形对象的位号名。

语法

图形对象名.TagName ="位号名"

举例

设置数据链接 DataLink1 的位号名为 NA001。

```
DataLink1.TagName = "NA001"
```

8.2.28 Text

Text 函数用于设置图形对象的文字内容。

语法

图形对象名.Text ="文字内容"

举例

设置按钮 Button1 的文字内容为 AAA。

```
Button1.Text= "AAA"
```

8.2.29 TextColor

TextColor 函数用于设置图形对象的文字颜色。

语法

图形对象名.TextColor =颜色值

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设置按钮 Button1 的文字颜色为红色。

```
Button1.TextColor= vbRed
```

8.2.30 TransparentColor

TransparentColor 函数用于设置图形对象的透明色。

语法

图形对象名.TransparentColor =颜色值


颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设置按钮 Button1 的透明色为红色。

```
Button1.TransparentColor = vbRed
```

8.3 “管道（Pipe）”专用接口

管道 Pipe 是指流程图中的  图形对象，在脚本中该图形对象默认名为“PipeN”，其中 N 为流程图中自动添加的序号。

管道 Pipe 的专用接口是指仅限管道 Pipe 应用的函数，包括 ColorInner、ColorOutter 和 PipeWidth。

8.3.1 ColorInner

ColorInner 函数用于设置管道内部的颜色。

语法

Pipe 名.ColorInner =颜色值

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设置管道 Pipe1 的内部颜色为红色。

```
Pipe1.ColorInner = vbRed
```

8.3.2 ColorOutter

ColorOutter 函数用于设置管道外部的颜色。

语法

Pipe 名.ColorOutter =颜色值

颜色值与具体颜色的对应关系请参见“颜色常数”。

举例

设置管道 Pipe1 的外部颜色为红色。

```
Pipe1.ColorOuter = vbRed
```

8.3.3 PipeWidth

PipeWidth 函数用于设置管道的宽度。

语法

Pipe 名.PipeWidth=宽度值


宽度值为整数，单位为像素。

举例

设置管道 Pipe1 的宽度为 4 像素。

```
Pipe1.PipeWidth= 4
```

8.4 “面板（Panel）”专用接口

面板 Panel 是指流程图中的图形对象，在脚本中该图形对象默认名为“PanelN”，其中 N 为流程图中自动添加的序号。面板 Panel 的专用接口是指仅限面板 Panel 应用的函数，包括 PanelStyle。

PanelStyle 函数用于设定面板的风格。

语法

面板名.PanelStyle=风格值


风格值是 0~9 的整数。

举例

设置面板 Panel1 的面板风格为凸起。

```
Panel1.PanelStyle = 8
```

8.5 “按钮（Button）”专用接口

按钮 Button 是指流程图中的图形对象，在脚本中该图形对象默认名为“ButtonN”，其中 N 为流程图中自动添加的序号。按钮 Button 的专用接口是指仅限按钮 Button 应用的函数，包括 ButtonStyle、IsManagedButton、IsSwitchButton 和 PicturePosition。

8.5.1 ButtonStyle

ButtonStyle 函数用于设置按钮的风格。

语法

按钮名.ButtonStyle=按钮风格值

按钮风格值与具体风格的对应关系请参见“按钮风格对照表”。

举例

设置按钮 Button1 的风格为椭圆形。

```
Button1.ButtonStyle= 2
```

8.5.2 IsManagedButton

IsManagedButton 函数用于设置按钮是否为受限按钮。

语法

按钮名.IsManagedButton =True 或 False

True 表示设定该按钮为受限按钮，False 表示设定该按钮为非受限按钮。

举例

设置按钮 Button1 为受限按钮。

```
Button1.IsManagedButton = True
```



提示：

设置为受限的按钮，只能由配置了“受限按钮权限”的用户来操作。受限按钮权限的配置是在监控用户授权软件中完成的，详细的配置方法请查看《监控用户授权软件》。

8.5.3 IsSwitchButton

IsSwitchButton 函数用于设置按钮是否为开关按钮。

语法

按钮名.IsSwitchButton=True 或 False

True 表示设定该按钮为开关按钮，False 表示设定该按钮为非开关按钮。

举例

设置按钮 Button1 为开关按钮。

```
Button1.IsSwitchButton= True
```



提示：

按钮被设置为开关按钮后，支持显示为按下和弹起两种样式。当开关量位号值为 ON 时按钮显示为按下状态。当开关量位号值为 OFF 时按钮显示为弹起状态。

8.5.4 PicturePosition

PicturePosition 函数用于设置按钮图片的显示位置。

语法

按钮名.PicturePosition = 位置值


位置值为 0~2 的整数。0 表示图片居中显示，1 表示图片靠左显示，2 表示图片靠右显示。

举例

设置按钮 Button1 的图片靠右显示。

```
Button1.PicturePosition = 2
```

8.6 “图片（Image）”专用接口

图片 Image 是指流程图中的图形对象，在脚本中该图形对象默认名为“ImageN”，其中 N 为流程图中自动添加的序号。图片 Image 的专用接口是指仅限面板 Image 应用的函数，包括 Opacity。

Opacity 函数用于设定图片的不透明度，图片对象的“是否透明”属性应否，此函数才有效。

语法

图片名.Opacity=不透明度值


不透明度值是 0~255 的整数。0 表示图片完全透明，255 表示图片完全不透明。

举例

设置图片 Image1 的不透明度为 50。

```
Image1.Opacity= 50
```

8.7 “组合框（RadioBox）”专用接口

组合框 RadioBox 是指流程图中的图形对象，在脚本中该图形对象默认名为“RadioBoxN”，其中 N 为流程图中自动添加的序号。组合框 RadioBox 的专用接口是指仅限组合框 RadioBox 应用的函数，包括 CurrentSelectedBtn 和 GetTagValue。

8.7.1 CurrentSelectedBtn

CurrentSelectedBtn 函数用于设定组合框的默认选择项目。

语法

组合框名.CurrentSelectedBtn = 默认选项序号

默认选项序号为 0 开始的整数，最大值为“选项数-1”。

举例

设置组合框 RadioBox1 的默认选项为第三个选项。

```
RadioBox1.CurrentSelectedBtn = 2
```

8.7.2 GetTagValue

GetTagValue 函数用于设定组合框的当前选择值。

语法

组合框名.GetTagValue vTag

返回值


vTag, 当前选中条目的编号。从 0 开始, 组合框中的第一个选项对应返回值 0。

举例

在流程图添加按钮 Button1 和组合框 RadioBox1, 按钮 Button1 后获取组合框 RadioBox1 中当前选项并显示到弹出框内。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
  DIM selIndex
  RadioBox1.GetTagValue selIndex
  msgbox selindex
  app.MsgBox selindex,"提示框", 0
End Sub
```

8.8 “菜单（Menu）”专用接口

菜单 Menu 是指流程图中的图形对象, 在脚本中该图形对象默认名为“MenuN”, 其中 N 为流程图中自动添加的序号。菜单 Menu 的专用接口是指仅限菜单 Menu 应用的函数, 包括 MenuStyle。

MenuStyle 函数用于设定菜单风格。

语法

菜单名.MenuStyle=风格值


风格值为 0 和 1。0 表示菜单为 3D 效果菜单, 1 表示菜单为平面菜单。

举例

设置菜单 Menu1 的风格为 3D 效果菜单。

```
Menu1.MenuStyle= 0
```

8.9 “定时器控件（Timer）”专用接口

定时器 Timer 是指流程图中的图形对象, 在脚本中该图形对象默认名为“TimerN”, 其中 N 为流程图中自动添加的序号。定时器 Timer 的专用接口是指仅限定定时器 Timer 应用的函数, 包括 Enable 和 Interval。

8.9.1 Enable

Enable 函数用于使能定时器。

语法

定时器名.Enable=True 或 False

输入参数

Ture 表示使能定时器，False 表示禁用定时器。

8.9.2 Interval

Interval 函数用于定时器的周期。

语法

定时器名.Interval=周期值


输入参数

周期值是整数形式，单位为毫秒。

8.9.3 举例

参见“OnTimer”。

8.10 “实时报警控件（Alarm）”专用接口

实时报警控件 Alarm 是指流程图中的图形对象，在脚本中该图形对象默认名为“AlarmN”，其中 N 为流程图中自动添加的序号。实时报警控件 Alarm 的专用接口是指仅限实时报警控件 Alarm 应用的函数，包括 AckCurList、AckOne 和 GetAlmCount 等。

8.10.1 AckCurList

AckCurList 函数用于确认当前屏的所有报警。

语法

实时报警控件名.AckCurList()

输入参数

无

返回值

无

举例

单击流程图中的按钮 Button1，确认实时报警控件 Alarm1 当前屏中的所有报警。

Dim str1

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    Alarm1.AckCurList()
End Sub
```

8.10.2 AckOne

AckOne 函数用于确认当前选中行的报警。

语法

实时报警控件名.AckOne()

输入参数

无

返回值

无

举例

单击流程图中的按钮 Button1，确认真时报警控件 Alarm1 当前选中行的报警。

```
Dim str1
Sub Button1_OnLButtonUp(nFlag, x, y)
    Alarm1.AckOne()
End Sub
```

8.10.3 GetAlmCount

GetAlmCount 函数用于获取实时报警控件中当前的报警数。

语法

实时报警控件名.GetAlmCount()

输入参数

无

返回值

长整型，实时报警控件中当前的报警数。

举例

单击流程图中的按钮 Button1，获取实时报警控件 Alarm1 中当前的报警数并将其显示到弹出框内。

```
Dim str6
Sub Button1_OnLButtonUp(nFlag, x, y)
```

```

    str6 =alarm1.GetAlmCount()
    MsgBox str6
End Sub

```

8.10.4 SetFilterTag

SetFilterTag 函数用于过滤实时报警控件中显示的报警。成功运行函数后，实时报警控件中仅显示指定位号的报警。

语法

实时报警控件名.SetFilterTag(strTagName)

输入参数

strTagName，字符串型位号名。

返回值

无

举例


单击流程图中的按钮 Button1，获取实时报警控件 Alarm1 中 Tag1 的报警数并将其显示到弹出框内。

```

Sub Button1_OnLButtonUp(nFlag, x, y)
    alarm1.SetFilterTag ("Tag1")
End Sub

```

8.11 “趋势控件（TrdGraph）”专用接口

趋势控件 TrdGraph 是指流程图中的图形对象，在脚本中该图形对象默认名为“TrdGraphN”，其中 N 为流程图中自动添加的序号。趋势控件 TrdGraph 的专用接口是指仅限趋势控件 TrdGraph 应用的函数，包括 AddPen、DeletePen 和 GetStdLineColor 等。

8.11.1 AddPen

AddPen 函数用于向趋势控件中添加画笔（不指定序号），与 InsertPen 类似。

语法

趋势控件名.AddPen(bstrTagName, color, nHiLimit, nLoLimit)

输入参数

bstrTagName，位号名。
color，线条颜色。
nHiLimit，坐标上限。

nLoLimit, 坐标下限。

返回值

成功与否, True 表示添加画笔成功, False 表示添加画笔失败。

举例

单击按钮 Button1, 则向趋势控件 TrdGraph1 中添加画笔。画笔关联的位号是 A2、颜色为绿色、坐标上下限为 100~0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.AddPen "A2", vbGreen, 100, 0
End Sub
```



提示：

AddPen 与 InsertPen 的区别是：

通过 AddPen 向控件中添加画笔时查找空位来添加画笔，当趋势控件中位号满时将不再执行函数，即不再添加画笔。

通过 InsertPen 向控件中添加画笔时添加画笔到指定的位置，原位置若已有画笔则作为原位置后一个画笔，若画笔已满则最后的画笔将溢出。

8.11.2 AddRDBPen

AddRDBPen 函数用于向趋势控件中添加关系库画笔，与 InsertRDBPen 类似。

语法

趋势控件名.AddRDBPen(bstrValueField, bstrQualityField, color, nHiLimit, nLoLimit)

参数

bstrValueField, 关系库位号的位号值字段。

bstrQualityField, 关系库位号的质量码字段。

color, 线条颜色。

nHiLimit, 坐标上限。

nLoLimit, 坐标下限。

返回值

成功与否, True 表示添加画笔成功, False 表示添加画笔失败。

举例

单击按钮 Button1, 则向趋势控件 TrdGraph1 中添加关系库画笔。画笔关联的位号值字段为 A2、质量码为 Good、颜色为绿色、坐标上下限为 100~0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.AddRDBPen "A2", "Good", vbGreen, 100, 0
```

End Sub

8.11.3 InsertPen

InsertPen 函数用于向趋势控件中添加画笔（指定序号），与 AddPen 类似。

语法

趋势控件名.InsertPen(nIndex, bstrTagName, color, nHiLimit, nLoLimit)

输入参数

nIndex，画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

bstrTagName，位号名。

color，线条颜色。

nHiLimit，坐标上限。

nLoLimit，坐标下限。

返回值

成功与否，True 表示添加画笔成功，False 表示添加画笔失败。

举例

单击按钮 Button1，则向趋势控件 TrdGraph1 中添加画笔 8。画笔关联的位号是 AI、颜色为绿色、坐标上下限为 100~0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.InsertPen 7, "AI", vbGreen, 100, 0
End Sub
```

8.11.4 InsertRDBPen

InsertRDBPen 函数用于向趋势控件中添加关系库画笔（指定序号），与 AddRDBPen 类似。

语法

趋势控件名.InsertRDBPen(nIndex, bstrValueField, bstrQualityField, color, nHiLimit, nLoLimit)

参数

nIndex，画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

bstrValueField，关系库位号的位号值字段。

bstrQualityField，关系库位号的质量码字段。

color，线条颜色。

nHiLimit，坐标上限。

nLoLimit，坐标下限。

返回值

成功与否，True 表示添加画笔成功，False 表示添加画笔失败。

举例

单击按钮 Button1，则向趋势控件 TrdGraph1 中添加关系库画笔 8。画笔关联的位号值字段是 A2、质量码为 Good、颜色为绿色、坐标上下限为 100~0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.InsertRDBPen 7, "A2", "Good", vbGreen, 100, 0
End Sub
```

8.11.5 DeletePen

DeletePen 函数用于删除趋势控件中指定序号的画笔。

语法

趋势控件名.DeletePen(nIndex)

输入参数

nIndex，画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

返回值

成功与否，True 表示删除画笔成功，False 表示删除画笔失败。

举例

单击按钮 Button1，删除趋势控件 TrdGraph1 中的画笔 8。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.DeletePen(7)
End Sub
```

8.11.6 SetPen

SetPen 函数用于修改趋势控件中指定画笔的信息。

语法

趋势控件名.SetPen(nIndex, bstrTagName, color, nHiLimit, nLoLimit)

输入参数

nIndex，画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

bstrTagName，位号名。

color，线条颜色。

nHiLimit，坐标上限。

nLoLimit，坐标下限。

返回值

成功与否，True 表示修改画笔成功，False 表示修改画笔失败。

举例

单击按钮 Button1，则修改趋势控件 TrdGraph1 中的画笔 8。修改后，画笔关联的位号是 AI、颜色为绿色、坐标上下限为 100~0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetPen 7, "AI", vbGreen, 100, 0
End Sub
```

8.11.7 SetRDBPen

SetRDBPen 函数用于修改趋势控件中指定序号的关系库画笔。

语法

趋势控件名.SetRDBPen(nIndex, bstrValueField, bstrQualityField, color, nHiLimit, nLoLimit)

输入参数

nIndex，画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

bstrValueField，关系库位号的位号值字段。

bstrQualityField，关系库位号的质量码字段。

color，线条颜色。

nHiLimit，坐标上限。

nLoLimit，坐标下限。

返回值

成功与否，True 表示修改画笔成功，False 表示修改画笔失败。

举例

单击按钮 Button1，则修改趋势控件 TrdGraph1 中的关系库画笔 8。修改后，画笔关联的位号值字段是 A2、质量码为 Good、颜色为绿色、坐标上下限为 100~0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetRDBPen 7, "A2", "Good", vbGreen, 100, 0
End Sub
```

8.11.8 GetPenCount

GetPenCount 函数用于获取趋势控件中画笔的数量。

语法

趋势控件名.GetPenCount()

输入参数

无

返回值

趋势控件中的画笔数。

举例

单击按钮 Button1，则获取趋势控件 TrdGraph1 的画笔数量，并将其显示在弹出框内。

```
Dim str6
Sub Button1_OnLButtonUp(nFlag, x, y)
    str6 = TrdGraph1.GetPenCount()
    MsgBox str6
End Sub
```

8.11.9 GetBackColor

GetBackColor 函数用于获取趋势控件的背景色。

语法

趋势控件名.GetBackColor()

输入参数

无

返回值

color，背景色颜色。

举例

获取趋势控件 TrdGraph1 的背景色。

```
TrdGraph1.GetBackColor()
```

8.11.10 SetBackColor

SetBackColor 函数用于设置趋势控件的背景色。

语法

趋势控件名.SetBackColor(color)

输入参数

color，背景色颜色。颜色值与具体颜色的对应关系请参见“颜色常数”。

返回值

无

举例

设置趋势控件 TrdGraph1 的背景色为红色。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetBackColor vbRed
End Sub
```

8.11.11 GetFrontColor

GetFrontColor 函数用于获取趋势控件的边框前景色。

语法

趋势控件名.GetFrontColor()

输入参数

无

返回值

color, 边框前景色颜色。

举例

获取趋势控件 TrdGraph1 的边框前景色。

```
TrdGraph1.GetFrontColor()
```

8.11.12 SetFrontColor

SetFrontColor 函数用于设置趋势控件的边框前景色。

语法

趋势控件名.SetFrontColor(color)

输入参数

color, 边框前景色颜色。颜色值与具体颜色的对应关系请参见“颜色常数”。

返回值

无

举例

设置趋势控件 TrdGraph1 的边框前景色为红色。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetFrontColor vbRed
End Sub
```

8.11.13 GetTextColor

GetTextColor 函数用于获取趋势控件的字体颜色。

语法

趋势控件名.GetTextColor()

输入参数

无

返回值

color, 字体颜色。

举例

获取趋势控件 TrdGraph1 的字体颜色。

```
TrdGraph1.GetTextColor()
```

8.11.14 SetTextColor

SetTextColor 函数用于设置趋势控件的字体颜色。

语法

趋势控件名.SetTextColor (color)

输入参数

color, 字体颜色。颜色值与具体颜色的对应关系请参见“颜色常数”。

返回值

无

举例

设置趋势控件 TrdGraph1 的字体颜色为红色。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetTextColor vbRed
End Sub
```

8.11.15 GetBarLineColor

GetBarLineColor 函数用于获取趋势控件的滑杆颜色。

语法

趋势控件名.GetBarLineColor()

输入参数

无

返回值

color, 滑杆颜色。

举例

获取趋势控件 TrdGraph1 的滑杆颜色。

```
TrdGraph1.GetBarLineColor()
```

8.11.16 SetBarLineColor

SetBarLineColor 函数用于设置趋势控件的滑杆颜色。

语法

趋势控件名.SetBarLineColor (color)

输入参数

color, 滑杆颜色。颜色值与具体颜色的对应关系请参见“颜色常数”。

返回值

无

举例

设置趋势控件 TrdGraph1 的滑杆颜色为红色。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetBarLineColor vbRed
End Sub
```

8.11.17 GetDisplayDateTime

GetDisplayDateTime 函数用于获取趋势控件的显示时间范围。

语法

趋势控件名.GetDisplayDateTime nBeginTime nEndTime

输入参数

无

返回值

nBeginTime, 趋势控件显示的起始时间, 精确到秒。

nEndTime, 趋势控件显示的结束时间, 精确到秒。

举例

单击按钮 Button1, 获取趋势控件 TrdGraph1 的显示时间并将起始时间显示到弹出框。

```
Dim T1
Dim T2
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.GetDisplayDateTime T1,T2
    app.MsgBox T1, "提示框", 0
End Sub
```

8.11.18 SetDisplayDateTime

SetDisplayDateTime 函数用于设置趋势控件的显示时间范围。

语法

趋势控件名.SetDisplayDateTime (bstrBeginTime bstrEndTime)

输入参数

nBeginTime, 趋势控件显示的起始时间。

nEndTime, 趋势控件显示的结束时间。

返回值

无

举例

单击按钮 Button1, 设置趋势控件 TrdGraph1 的显示时间为"2015-10-09 10:00:00"至"2015-10-09 12:00:00"。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetDisplayDateTime "2015-10-09 10:00:00","2015-10-09 12:00:00"
End Sub
```

8.11.19 ShowStdLines

ShowStdLines 函数用于设置在趋势控件中显示标准限。

语法

趋势控件名.ShowStdLines()

输入参数

无

返回值

无

举例

单击按钮 Button1，设置趋势控件 TrdGraph1 显示标准限。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.ShowStdLines()
End Sub
```

8.11.20 HideStdLines

HideStdLines 函数用于设置在趋势控件中隐藏标准限。

语法

趋势控件名.HideStdLines()

输入参数

无

返回值

无

举例

单击按钮 Button1，设置趋势控件 TrdGraph1 隐藏标准限。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.HideStdLines()
End Sub
```

8.11.21 SetRDBInfo

SetRDBInfo 函数用于设置趋势控件中的关系库数据源及表名。

语法

趋势控件名.SetRDBInfo(bstrRDBSource, bstrTableName)

输入参数

bstrRDBSource，关系库数据源。

bstrTableName，关系库表名。

返回值

无

举例

单击按钮 Button1，则设置趋势控件 TrdGraph1 的关系库数据源为“172.30.0.130/SCADA”、表名为“test”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetRDBPen "172.30.0.130/SCADA", "test"
End Sub
```

8.11.22 SetRDBTimeField

SetRDBTimeField 函数用于设置趋势控件中的关系库位号的时间字段。

语法

趋势控件名.SetRDBTimeField(bstrTimeFieldName)

输入参数

bstrTimeFieldName，关系库位号的时间字段。

返回值

无

举例

单击按钮 Button1，则设置趋势控件 TrdGraph1 的关系库位号的时间字段为“2015-10-09 10:00:00”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetRDBTimeField "2015-10-09 10:00:00"
End Sub
```

8.11.23 Refresh

Refresh 函数用于刷新趋势控件的显示。

语法

趋势控件名.Refresh()

输入参数

无

返回值

无

举例

单击按钮 Button1，则刷新趋势控件 TrdGraph1 的显示。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.Refresh()
End Sub
```

8.11.24 SetStdLineColor

SetStdLineColor 函数用于设置趋势控件中的标准限颜色。

语法

趋势控件名.SetStdLineColor (color)

输入参数

color，颜色值。颜色值与具体颜色的关系请参见“颜色常数”。

返回值

无

举例

单击按钮 Button1，则设置趋势控件 TrdGraph1 的标准限颜色为红色。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetStdLineColor (vbRed)
End Sub
```

8.11.25 GetStdLineColor

GetStdLineColor 函数用于获取趋势控件中的标准限颜色。

语法

趋势控件名.GetStdLineColor()

输入参数

无

返回值

color，颜色值。颜色值与具体颜色的关系请参见“颜色常数”。

举例

单击按钮 Button1，则获取趋势控件 TrdGraph1 的标准限颜色为红色，并将其显示在提示框。

```
DIM C1
```

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    C1=TrdGraph1.GetStdLineColor()
    app.MsgBox C1, "提示框", 0
End Sub
```

8.11.26 GetStdLineShowMode

GetStdLineShowMode 函数用于获取趋势控件中的标准限模式。

语法

趋势控件名.GetStdLineShowMode()

输入参数

无

返回值

返回值为 0 表示标准限模式为“高限”，为 1 表示标准限模式为“高高限”，为 2 表示标准限模式为“自定义”。

举例

单击按钮 Button1，则获取趋势控件 TrdGraph1 的标准限模式，并将其显示在提示框内。

```
DIM C1
Sub Button1_OnLButtonUp(nFlag, x, y)
    C1=TrdGraph1.GetStdLineShowMode()
    app.MsgBox C1, "提示框", 0
End Sub
```

8.11.27 SetStdLineShowMode

SetStdLineShowMode 函数用于设置趋势控件中的自定义标准限的高低限。

语法

趋势控件名.SetStdLineShowMode(nMode, nUserDefineUp, nUserDefineLow)

输入参数

nMode，标准限模式。必须设置为 2，否则后续参数无效。

nUserDefineUp，自定义高限。

nUserDefineLow，自定义低限。

返回值

无

举例

单击按钮 Button1，则设置趋势控件 TrdGraph1 的自定义高低限分别为 100 和 0。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetStdLineShowMode 2,100,0
End Sub
```

8.11.28 SetTrdShowType

SetTrdShowType 函数用于设置趋势控件的数据显示类型。

语法

趋势控件名.SetTrdShowType (nType)

输入参数

nType，趋势数据显示类型。为 0 表示百分量，为 1 表示工程量。

返回值

无

举例

单击按钮 Button1，则设置趋势控件 TrdGraph1 的数据显示类型为百分量显示。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    TrdGraph1.SetTrdShowType 0
End Sub
```

8.11.29 GetTrdShowType

GetTrdShowType 函数用于获取趋势控件的数据显示类型。

语法

趋势控件名.GetTrdShowType()

输入参数

无

返回值

nType，趋势数据显示类型。为 0 表示百分量，为 1 表示工程量。

举例

单击按钮 Button1，则获取趋势控件 TrdGraph1 的数据显示类型，并将其显示在提示框。

```
DIM C1
```

```

Sub Button1_OnLButtonUp(nFlag, x, y)
    C1=TrdGraph1.GetTrdShowType()
    app.MsgBox C1, "提示框", 0
End Sub

```

8.11.30 GetPen

GetPen 函数用于获取趋势控件中指定序号的画笔信息。

语法

趋势控件名.GetPen(nIndex, bstrTagName, color, nHiLimit, nLoLimit)

输入参数

nIndex, 画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

返回值

bstrTagName, 位号名。

color, 线条颜色。

nHiLimit, 坐标上限。

nLoLimit, 坐标下限。

举例

单击按钮 Button1, 则获取趋势控件 TrdGraph1 画笔 1 的信息, 并将其显示在提示框。

```
DIM C1
```

```
DIM C2
```

```
DIM C3
```

```
DIM C4
```

```
Sub Button1_OnLButtonUp(nFlag, x, y)
```

```
    TrdGraph1.GetPen 0,C1,C2,C3,C4
```

App.MsgBox "位号: "&C1& chr(10) & "画笔颜色: " &C2& chr(10), "Value",0 '两个内容之间加"chr(10)"表示分行显示

8.11.31 GetRDBPen

GetRDBPen 函数用于获取趋势控件中指定的关系库画笔信息。

语法

趋势控件名.GetRDBPen(nIndex, bstrValueField, bstrQualityField, color, nHiLimit, nLoLimit)

输入参数

nIndex, 画笔序号。0~7 的长整型对应趋势控件“属性”弹出对话框的“画笔 1”~“画笔 8”。

返回值

bstrValueField, 关系库位号的位号值字段。

bstrQualityField, 关系库位号的质量码字段。

color, 线条颜色。

nHiLimit, 坐标上限。

nLoLimit, 坐标下限。

举例

单击按钮 Button1, 则获取趋势控件 TrdGraph1 画笔 1 的信息, 并将其显示在提示框。

```
DIM C1
```

```
DIM C2
```

```
DIM C3
```

```
DIM C4
```

```
DIM C5
```


```
Sub Button1_OnLButtonUp(nFlag, x, y)
```

```
    TrdGraph1.GetRDBPen 0,C1,C2,C3,C4,C5
```

```
    App.MsgBox "关系库的位号值: "&C1& chr(10) & "关系库位号的质量码: " &C2& chr(10),
```

"Value",0 '两个内容之间加"chr(10)"表示分行显示

8.12 下拉框控件（ComboBox）专用接口

下拉框 ComboBox 是指流程图中的  图形对象, 在脚本中该图形对象默认名为“ComboBoxN”, 其中 N 为流程图中自动添加的序号。下拉框 ComboBox 的专用接口是指仅限下拉框 ComboBox 应用的函数, 包括 AddItem、RemoveItem 和 AddString 等。

8.12.1 下拉框 ComboBox 的事件函数 Change

Change 函数用于定义下拉框中选择项变化的事件。

语法

下拉框名_change

输入参数

无

举例

当下拉框 ComBoBox1 中的选择项变化时, 弹出消息框“选择项变化”。

```
Sub ComBoBox1_Change()
```

```
    msgbox("选择项发生变化")
```

```
End Sub
```

8.12.2 AddItem

AddItem 函数用于向下拉框添加指定序号的选项, 与 AddString 类似。

语法

下拉框名.AddItem(nIndex,lpszString)

输入参数

nIndex 为从 0 开始的短整型数，用于表示下拉列表索引号。当 nIndex 为 0 时，对应的为下拉框的第一个选项。以此类推，nIndex 为 N，对应下拉框的 N+1 选项。

lpszString 为字符串型，用于表示添加的数据。

返回值

为长整型值。

举例

按下按钮“Button1”，向下拉框 ComboBox1 的第 1 项中添加数据“a”。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    Combobox1.AddItem 0,"a"
End Sub
```



提示：

- 若为全新写值，即原下拉框中没有任何选项，则必须从 0 开始按顺序对下拉列表进行写值。
- 若对已有内容的选项进行写值，则新增内容插入到对应索引项中，原索引项及其后选项中的内容将按原顺序后延一位。

8.12.3 AddString

AddString 函数用于向下拉框不带序号的添加数据，与 AddItem 类似。

语法

下拉框名.AddString (lpszString)

输入参数

lpszString 为字符串型，用于表示添加的数据。

返回值

为长整型值。

举例

按下按钮“add1”，向下拉框 ComboBox1 中添加数据“a”。

```
Sub add1_OnLButtonUp(nFlag, x, y)
    ComboBox1.AddString "a"
End Sub
```

8.12.4 Clear

Clear 函数用于清空下拉框中的数据。

语法

下拉框名.Clear

输入参数

无

举例

按下按钮 Button1，则清空下拉框 ComboBox1 中的数据。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    ComboBox1.Clear
End Sub
```

8.12.5 FindString

FindString 从指定位置开始查找某数据所在的索引项。

语法

下拉框名.FindString

输入参数

nStartAfter: 开始查询的下拉列表索引（短整型）
lpszString: 查找位置的数据（字符串型）

返回值

长整型

举例

弹出的信息框将显示数据”aa”所在的下拉框的索引项

```
Sub search_str_OnLButtonUp(nFlag, x, y)
    msgbox "ComboBox:" & ComboBox1.FindString(1,"aa")
End Sub
```

8.12.6 GetCount

GetCount 函数用于获取下拉框中的选项个数。

语法

下拉框名.GetCount

输入参数

无

举例

按下按钮 Button1，则弹出信息框显示下拉框列表中的数据个数。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
str2 = Combobox1.GetCount
Msgbox "ComboBox:" & str2
End Sub
```

8.12.7 GetCurSel

GetCurSel 函数用于获取下拉框当前选中项的索引值。

语法

下拉框名.GetCurSel

输入参数

无

返回值

长整型，当前选中项的索引值。返回值为-1 则表示未选中任何选项。

举例

按下按钮 Button1，则获取下拉框 ComboBox1 当前选中项的索引值并将其显示在弹出框内。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    str2 = Combobox1.GetCurSel
    MsgBox "ComboBox:" & str2
End Sub
```

8.12.8 GetCurText

GetCurText 函数用于获取下拉框当前选中项的内容。

语法

下拉框名.GetCurText

输入参数

无

返回值

字符串型，当前选中项的内容。

举例

按下按钮 Button1，则获取下拉框 ComboBox1 当前选中项的内容并将其显示在弹出框内。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    str2 = Combobox1.GetCurText
    MsgBox "ComboBox:" & str2
End Sub
```

8.12.9 RemoveItem

RemoveItem 函数用于删除下拉框指定索引值的选项。

语法

下拉框名.RemoveItem(pvargIndex)

输入参数

pvargIndex 为从 0 开始的短整型数，用于表示下拉列表索引号。当 pvargIndex 为 0 时，对应的为下拉框的第一个选项。以此类推，pvargIndex 为 N，对应下拉框的 N+1 选项。

返回值

长整型。

举例

按下按钮 Button1，则删除下拉框 ComboBox1 的第二项。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    Combobox1.RemoveItem 1
End Sub
```

8.12.10 SetCurSel

SetCurSel 函数用于选中下拉框中指定选项。

语法

下拉框名.SetCurSel (nSelect)

输入参数

nSelect 为从 0 开始的短整型数，用于表示下拉列表索引号。当 nSelect 为 0 时，对应的为下拉框的第一个选项。以此类推，nSelect 为 N，对应下拉框的 N+1 选项。

返回值


长整型。

举例

按下按钮 Button1，则选中下拉框 ComboBox1 的第二项。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    Combobox1.SetCurSel 1
End Sub
```

8.13 日期选择控件 DateTimePicker 专用接口

日期选择控件 DateTimePicker 是指流程图中的  图形对象，在脚本中该图形对象默认名为“DateTimePickerN”，其中 N 为流程图中自动添加的序号。日期选择控件 DateTimePicker 的专用接口是指仅限日期选择控件 DateTimePicker 应用的函数，包括 GetDay、GetDayOfWeek 和 GetMonth 等。

8.13.1 GetDay

GetDay 函数用于获取当前日期选择控件中显示的日期。

语法

日期选择控件名.GetDay

输入参数

无

返回值

长整型值，当前显示的日期，即几号。

举例

单击流程图中的按钮 Button1，将弹出信息框显示“今天是 str1 号”，str1 为 DateTimePicker1 中选中的日期。

```
Dim str1
Sub Button1_OnLButtonUp(nFlag, x, y)
    str1 = DateTimePicker1.GetDay
    MsgBox " 今天是" & str1 & "号"
End Sub
```

8.13.2 GetDayOfWeek

GetDayOfWeek 函数用于获取日期选择框中当前显示日期是该星期的第几天(星期日为第一天)。

语法

日期选择控件名.GetDayOfWeek

输入参数

无

返回值

长整型值，当前显示日期是该星期的第几天。1 表示星期日、2 表示星期一，以此类推 7 表示星期六。

举例

按下按钮 Button2 时，弹出信息框显示日期选择控件 DateTimePicker1 当前显示的日期是星期几。

```
Sub Button2_OnLButtonUp(nFlag, x, y)
    If DateTimePicker1.GetDayOfWeek = "1" Then
        MsgBox " 星期天"
    ElseIf DateTimePicker1.GetDayOfWeek = "2" Then
        MsgBox " 星期一"
    ElseIf DateTimePicker1.GetDayOfWeek = "3" Then
        MsgBox " 星期二"
    ElseIf DateTimePicker1.GetDayOfWeek = "4" Then
        MsgBox " 星期三"
    ElseIf DateTimePicker1.GetDayOfWeek = "5" Then
        MsgBox " 星期四"
    ElseIf DateTimePicker1.GetDayOfWeek = "6" Then
        MsgBox " 星期五"
    ElseIf DateTimePicker1.GetDayOfWeek = "7" Then
        MsgBox " 星期六"
    End If
End Sub
```

8.13.3 GetDayOfYear

GetDayOfYear 函数用于获取当前显示的日期是该年的第几天。

语法

日期选择控件名.GetDayOfYear

输入参数

无

返回值

长整型值，当前显示日期是该年中的第几天。

举例

按下按钮 Button3 时，弹出信息框显示日期选择控件 DateTimePicker1 当前显示的日期是哪一年中的哪一天。

```
Dim str3
Dim str8
Sub Button3_OnLButtonUp(nFlag, x, y)
    str3 = DateTimePicker1.GetDayOfYear
    str8 = DateTimePicker1.GetYear
    MsgBox " 今天是"& str8 &"年的第" & str3 &"天"
End Sub
```

8.13.4 GetYear

GetYear 函数用于获取当前显示的日期是哪一年。

语法

日期选择控件名.GetYear

输入参数

无

返回值

长整型值，当前显示日期的年份。

举例

按下按钮 Button3 时，弹出信息框显示日期选择控件 DateTimePicker1 当前显示的日期是哪一年中的哪一天。

```
Dim str3
Dim str8
Sub Button3_OnLButtonUp(nFlag, x, y)
    str3 = DateTimePicker1.GetDayOfYear
    str8 = DateTimePicker1.GetYear
    MsgBox " 今天是"& str8 &"年的第" & str3 &"天"
End Sub
```

8.13.5 GetHour

GetHour 函数用于获取当前显示的小时。

语法

日期选择控件名.GetHour

输入参数

无

返回值

当前显示的小时，为 0~23 的长整型数。

举例

当按下按钮 Button4 时，弹出信息框显示当前具体时间，精确到秒。

```
Dim str4
Dim str5
Dim str7
Sub Button4_OnLButtonUp(nFlag, x, y)
    str4 = DateTimePicker1.GetHour
    str5 = DateTimePicker1.GetMinute
    str7 = DateTimePicker1.GetSecond
    MsgBox str4 & "点" & str5 & "分" & str7 & "秒"
End Sub
```

8.13.6 GetMinute

GetMinute 函数用于获取当前显示的分钟。

语法

日期选择控件名.GetMinute

输入参数

无

返回值

当前显示的分钟，为 0~59 的长整型数。

举例

当按下按钮 Button4 时，弹出信息框显示当前具体时间，精确到秒。

```
Dim str4
Dim str5
Dim str7
Sub Button4_OnLButtonUp(nFlag, x, y)
    str4 = DateTimePicker1.GetHour
    str5 = DateTimePicker1.GetMinute
    str7 = DateTimePicker1.GetSecond
    MsgBox str4 & "点" & str5 & "分" & str7 & "秒"
End Sub
```

8.13.7 GetSecond

GetSecond 函数用于获取当前显示的秒。

语法

日期选择控件名.GetSecond

输入参数

无

返回值

当前显示的秒，为 0～59 的长整型数。

举例

当按下按钮 Button4 时，弹出信息框显示当前具体时间，精确到秒。

```
Dim str4
Dim str5
Dim str7
Sub Button4_OnLButtonUp(nFlag, x, y)
    str4 = DateTimePicker1.GetHour
    str5 = DateTimePicker1.GetMinute
    str7 = DateTimePicker1.GetSecond
    MsgBox str4 & "点" & str5 & "分" & str7 & "秒"
End Sub
```

8.13.8 GetMonth

GetMonth 函数用于获取当前显示的月份。

语法

日期选择控件名.GetMonth

输入参数

无

返回值


当前显示的月份，为 1～12 的长整型数。

举例

按下按钮 Button6，弹出信息框显示 DateTimePicker1 当前显示的月份。

```
Dim str6
Sub Button6_OnLButtonUp(nFlag, x, y)
    str6 = DateTimePicker1.GetMonth
    MsgBox str6 & "月"
End Sub
```

8.14 树形控件 WebTreeView 专用接口

树形控件 WebTreeView 是指流程图中的  图形对象，在脚本中该图形对象默认名为“WebTreeViewN”，其中 N 为流程图中自动添加的序号。树形控件 WebTreeView 的专用接口是指仅限树形控件 WebTreeView 应用的函数，包括 AddNode、CheckAll 和 DeleteAllNodes 等。

8.14.1 树形控件 WebTreeView 的事件函数 OnClick

OnClick 函数用于定义树形控件中选中指定节点的事件。

语法

树形控件名_OnClick，如：

```
Sub WebTreeView1_OnClick(szValue,szText,nType)
End Sub
```

输入参数

szValue，字符串类型，节点值。

szText：字符串类型，节点显示的文本。

nType：数值形式，用户自定义的节点类型。0 表示根节点，1 表示子节点。

举例

```
Sub WebTreeView1_OnClick(szValue,szText,nType)
    txtText.Text = szText
    txtValue.Text = szValue
    txtType.Text= cstr (nType)
End Sub
```

8.14.2 AddNode

AddNode 函数向树形控件中添加节点。

语法

WebTreeView 名.AddNode(lpcstrValue,lpcstrText,nType,nPid, bCheckEnable)

输入参数

lpcstrValue：字符串类型，节点值。

lpcstrText：字符串类型，节点显示的文本。

nType：数值形式，用户自定义的节点类型。

nPid：数值形式，该节点所属的父节点编号。当填写不存在的节点编号时，该树形节点会被添加为根节点，否则添加为指定节点的子节点。

bCheckEnable：布尔类型，是否显示复选框。

返回值

长整型，添加节点的序号。

举例

单击按钮 Button1，则向 WebTreeView 中添加根节点“江苏”。单击按钮 Button4，则向“江苏”根节点中添加子节点“南京”、“苏州”、“扬州”和“无锡”。

```
DIM js_id
Sub Button1_OnLButtonUp(nFlag, x, y)
    js_id= WebTreeView1.AddNode ("js","江苏",0,-1,True)
    app.MsgBox js_id,"提示框", 0
End Sub
Sub Button4_OnLButtonUp(nFlag, x, y)
    WebTreeView1.AddNode "nanjin","南京",1,js_id,True
    WebTreeView1.AddNode "suzhou","苏州",1,js_id,True
    WebTreeView1.AddNode "yangzhou","扬州",1,js_id,True
    WebTreeView1.AddNode "wuxi","无锡",1,js_id,True
End Sub
```

8.14.3 CheckAll

CheckAll 函数用于选中或取消选中树形控件的所有节点。

语法

WebTreeView 名.CheckAll(bCheck)

输入参数

bCheck: BOOL 型，用于表示是否全选所有节点。True 表示全选所有节点，False 表示取消全选所有节点。

返回值

无

举例

单击按钮 Button1，则选中 WebTreeView 中的所有节点。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    WebTreeView1.CheckAll (true)
End Sub
```

8.14.4 DeleteAllNodes

DeleteAllNodes 函数用于删除树形控件的所有节点。

语法

WebTreeView 名.DeleteAllNodes

输入参数

无

返回值

BOOL，表示是否操作成功。True 表示操作成功，False 表示操作失败。

举例

单击按钮 Button1，则删除 WebTreeView 中的所有节点。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    WebTreeView1.DeleteAllNodes
End Sub
```

8.14.5 DeleteNode

DeleteNode 函数用于删除树形控件的指定节点。

语法

WebTreeView 名.DeleteNode(nNodeId)

输入参数

nNodeId，长整型，节点编号。

返回值

BOOL，表示是否操作成功。True 表示操作成功，False 表示操作失败。

举例

单击按钮 Button1，则删除 WebTreeView 中的节点 5。

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    WebTreeView1.DeleteNode(5)
End Sub
```

8.14.6 GetCheckedLeafNodes

GetCheckedLeafNodes 函数用于获取树形控件的所有已选的叶节点。

语法

WebTreeView 名.GetCheckedLeafNodes(pvValue,pvText,pvType)

输出参数

pvValue: 节点值数组。
 pvText: 节点文本数组。
 pvType: 节点类型数组。

返回值

已选的叶节点数。

举例

单击按钮 Button1，则获取 WebTreeView 中当前选中叶节点的信息，并将选中的节点显示在提示框。

```
Dim n
Dim pvValue
Dim pvText
Dim pvType
Sub Button4_OnLButtonUp(nFlag, x, y)
    n = WebTreeView1.GetCheckedLeafNodes (pvValue,pvText,pvType)
    For i=0 To n-1
        app.msgbox pvValue(i)&"-"&pvText(i)&"-"&pvType(i),"TIPS",0
    Next
End Sub
```

8.14.7 GetCheckedNodes

GetCheckedNodes 函数用于获取树形控件的所有已选的节点，包括叶节点和根节点。

语法

WebTreeView 名.GetCheckedNodes(pvValue,pvText, pvType)

输出参数

pvValue: 节点值数组。
 pvText: 节点文本数组。
 pvType: 节点类型数组。

返回值

已选的节点数，包括叶节点和根节点。

举例

单击按钮 Button1，则获取 WebTreeView 中当前选中节点的信息，并将选中的节点显示在提示框。

```
Dim n
```

```

Dim pvValue
Dim pvText
Dim pvType
Sub Button4_OnLButtonUp(nFlag, x, y)
    n = WebTreeView1.GetCheckedNodes (pvValue,pvText,pvType)
    For i=0 To n-1
        app.msgbox pvValue(i)&"-"&pvText(i)&"-"&pvType(i),"TIPS",0
    Next
End Sub

```

8.14.8 Inverse

Inverse 函数用于反选树形控件中已选的节点。

语法

WebTreeView 名.Inverse

输入参数

无

返回值

无

举例

单击按钮 Button4，则反选 WebTreeView 中当前选中节点。

```

Sub Button4_OnLButtonUp(nFlag, x, y)
    WebTreeView1.Inverse
End Sub

```

8.14.9 UpdateNode

UpdateNode 函数用于修改树形控件中指定的节点。

语法

WebTreeView 名.UpdateNode(nNodeId,lpcstrValue, lpcstrText, nType)

输入参数

nNodeId: 需要更新节点的 ID 号。

lpcstrValue: 字符串类型，节点值。

lpcstrText: 字符串类型，节点显示的文本。

nType: 数值形式，用户自定义的节点类型。

返回值


BOOL 型，表示是否操作成功。True 表示操作成功，False 表示操作失败。

举例

单击按钮 Button1，则修改 WebTreeView 中 2 号节点为“常州”。

```
Sub Button4_OnLButtonUp(nFlag, x, y)
    WebTreeView1.UpdateNode 1"changzhou","常州",1
End Sub
```

8.15 数据库表控件 DataBase 专用接口

数据库表控件 DataBase 是指流程图中的  图形对象，在脚本中该图形对象默认名为“DataBaseN”，其中 N 为流程图中自动添加的序号。数据库表控件 DataBase 的专用接口是指仅限数据库表控件 DataBase 应用的函数，包括 CreateListColumn、AddDataToListCtrl、OnExport、QueryByField、ClearListCtrl、ClearQueryCondition 和 QueryBySql。

8.15.1 CreateListColumn

CreateListColumn 函数用于在数据库中创建表格。

语法

DataBase 名.CreateListColumn vCloumns,width

输入参数

vCloumns: 列名（变体型）。

width: 表格列宽（LONG 型）。

返回值

LONG 型，表示是否操作成功。0 表示操作成功，1 表示操作失败。

举例

```
Sub Button1_OnLButtonDown(nFlag, x, y)
    Dim cloumns(5),width
    cloumns(0) = "日期"
    cloumns(1) = "时间"
    cloumns(2) = "班次"
    cloumns(3) = "起始"
    cloumns(4) = "终止"
    cloumns(5) = "操作人"
    width = 100
    DataBase1.CreateListColumn cloumns,width
End Sub
```

8.15.2 AddDataToListCtrl

AddDataToListCtrl 函数用于按行添加表格数据。

语法

DataBase 名.AddDataToListCtrl vRows

输入参数

vRows: 列名（变体型）。

返回值

LONG 型，表示是否操作成功。0 表示操作成功，1 表示操作失败。

举例

```
Sub Button2_OnLButtonDown(nFlag, x, y)
    Dim row(5)
    row(0) = "2022-11-15"
    row(1) = "14:55"
    row(2) = "123456"
    row(3) = "西安"
    row(4) = "北京"
    Dim user
    user = App.GetCurrentUser
    row(5) = user
    DataBase1.AddDataToListCtrl row
End Sub
```

8.15.3 OnExport

OnExport 函数用于导出当前屏函数到指定路径。

语法

DataBase 名.OnExport path

输入参数

path: 导出数据的指定路径（字符串型）。

返回值

LONG 型，表示是否操作成功。0 表示操作成功，1 表示操作失败。

举例

```
Sub Button4_OnLButtonDown(nFlag, x, y)
```

```
Dim path
path="C:\3.csv"
DataBase1.OnExport path
End Sub
```

8.15.4 QueryByField

QueryByField 函数用于根据字段查询数据库的数据。

语法

DataBase 名.QueryByField fields,queryBegin,queryEnd

输入参数

fields: 需要过滤的列名称，必须和表格控件上显示的一致（字符串型）。

queryBegin: 查询的起始条件（字符串型）。

queryEnd: 查询的结束条件（字符串型）。

返回值

LONG 型，表示是否操作成功。0 表示操作成功，1 表示操作失败。

举例

```
Sub Button5_OnLButtonDown(nFlag, x, y)
Dim fields,queryBegin,queryEnd
fields = "起始"
queryBegin= "西安"
queryEnd= "西安"
DataBase1.QueryByField fields,queryBegin,queryEnd
End Sub
```

8.15.5 ClearListCtrl

ClearListCtrl 函数用于清空当前表格控件上的所有内容。

语法

DataBase 名.ClearListCtrl

输入参数

无

返回值

无

举例

```
Sub Button3_OnLButtonDown(nFlag, x, y)
    DataBase1.ClearListCtrl
End Sub
```

8.15.6 ClearQueryCondition

ClearQueryCondition 函数用于清空查询条件。仅适用于通过 QueryByField 函数查询的数据。

语法

DataBase 名.ClearQueryCondition

输入参数

无

返回值

无

举例

```
Sub Button6_OnLButtonDown(nFlag, x, y)
    DataBase1.ClearQueryCondition
End Sub
```

8.15.7 QueryBySql

通过 QueryBySql 函数, 可以根据 sql 语句查询数据。使用 QueryBySql 函数时, 组态期控件中“已经选中的列”中的列必须是 sql 语句中 select 后输入的列的子集, 否则会查询失败。

语法

DataBase 名. QueryBySql sql

输入参数

sql: 需要查询的 SQL 语句（字符串型）。

返回值

LONG 型, 表示是否操作成功。0 表示操作成功, 1 表示操作失败。

举例

```
Sub Button7_OnLButtonDown(nFlag, x, y)
    Dim sql
    sql="select 3fdf,44S,AA,col1,col2,col3,DD,FV,QC,SS,time1 from test limit 1,10;"
    DataBase1.QueryBySql sql
```

End Sub

9 事件触发函数

事件触发指当某事件发生后，触发某函数执行。

流程图脚本中支持下述所有事件，调度脚本支持的事件请查看 10 调度支持的脚本。

9.1 OnLButtonUp

鼠标左键放开事件。当鼠标左键从图形对象上放开后，函数被执行。

例：鼠标左键从按钮上放开时，圆角矩形顺时针旋转 60 度。

在流程图中画一个按钮（名称为 **Button1**）和一个圆角矩形（名称为 **RoundRect1**），编辑脚本如下：

```
Sub Button1_OnLButtonUp(nFlag, x, y)
    RoundRect1.Angle = 60
End Sub
```

在监控中左键点击按钮（**Button1**）放开后，圆角矩形（**RoundRect1**）顺时针旋转 60 度。

9.2 OnLButtonDown

鼠标左键点击事件，当鼠标左键点击图形对象后该事件被执行。

例：鼠标左键点击按钮时，矩形顺时针旋转 30 度。

在流程图中画一个按钮（名称为 **Button1**）和矩形（名称为 **Rect1**），编辑脚本如下：

```
Sub Button1_OnLButtonDown(nFlag, x, y)
    Rect1.Angle = 30
End Sub
```

在监控中左键点击按钮（**Button1**）时，矩形（**Rect1**）顺时针旋转 30 度。

9.3 OnLButtonDblClk

鼠标左键双击事件。当鼠标左键双击图形对象后，函数被执行。

例：鼠标左键双击按钮时椭圆下边界值改为 200。

在流程图中画一个按钮（名称为 **Button1**）和椭圆（名称为 **Ellipse1**），编辑脚本如下：

```
Sub Button1_OnLButtonDblClk(nFlag, x, y)
    Ellipse1.Bottom = 200
End Sub
```

在监控中左键双击按钮（**Button1**），椭圆（**Ellipse1**）下边界置为 200。

9.4 OnRButtonClk

鼠标右键单击事件。当鼠标右键单击图形对象后，函数被执行。

例：鼠标右键单击按钮后管道置图层 2。

在流程图中画一个按钮（名称为 **Button1**）和管道（名称为 **Pipe1**），编辑脚本如下：

```
Sub Button1_OnRButtonClk(nFlag, x, y)
    Pipe1.Layer = 2
End Sub
```

在监控中鼠标右键单击按钮（**Button1**），管道（**Pipe1**）的图层属性置为 2。

9.5 OnMouseMove

鼠标移动事件。当鼠标在图形对象中移动时，函数被执行。

例：鼠标在按钮上移动时多边形宽度加 5。

在流程图中画一个按钮（名称为 **Button1**）和多边形（名称为 **Polygon1**），编辑脚本如下：

```
Sub Button1_OnMouseMove(nFlag, x, y)
    Polygon1.Width = Polygon1.Width + 5
End Sub
```

在监控中鼠标在按钮（**Button1**）上移动一次时，多边形（**Polygon1**）的宽度加 5。

9.6 OnMouseEnter

鼠标移入事件。当鼠标光标移动到图形对象中时，函数被执行。

例：当鼠标移入按钮时，将扇形隐藏。

在流程图中画一个按钮（名称为 **Button1**）和扇形（名称为 **Pie1**），编辑脚本如下：

```
Sub Button1_OnMouseEnter()
    Pie1.Visible = False
End Sub
```

在监控中将鼠标光标移入按钮（**Button1**）区域后，扇形（**Pie1**）被隐藏。

9.7 OnMouseLeave

鼠标移出事件。当鼠标从图形对象中移出时，函数被执行。

例：当鼠标移出按钮时，文字居中。

在流程图中画一个按钮（名称为 **Button1**）并添加一个文字图形对象（名称为 **Text1**），编辑脚本如下：

```
Sub Button1_OnMouseLeave()
    Text1.Align = 2
End Sub
```

在监控中将鼠标移出按钮（**Button1**）时，文字图形对象的文字居中（**Text1**）。

9.8 OnTimer

当图形对象按钮点击后使定时器定时功能有效。

在画面中画一个按钮（名称为 **Button1**）和定时器（名称为 **Timer1**），编辑脚本如下：

```

Sub Timer1_OnTimer()
Text2.Text=Now()
End Sub

Sub Button1_OnLButtonUp(nFlag, x, y)
    i=3000
Timer1.Interval=i '这里的 3000 是以毫秒为单位
If Button1.Text="停止计时" Then
    Timer1.Enable=False
    Button1.Text="开始计时"
Else
    Timer1.Enable=True
    Button1.Text="停止计时"
End If
End Sub

```

当脚本被执行后，i=3000 时，每隔 3 秒钟 Text2 显示一下当前时间。可以通过改变 i 的值来改变事件的触发间隔。

9.9 onDataChange

Datalink 值变化事件。当 Datalink 值变化时，函数被执行。

例如：在位号值发生改变时，给出提示。编辑脚本如下：

```

Sub DataLink1_OnDataChange(val)
    App.MsgBox "值在改变", "SCADA", 0
End Sub

```

组态期：流程图中添加 DataLink，关联一个开关量位号。监控期：当开关量位号变化时，将弹出标题栏为“SCADA”，内容为“值在改变”的提示框。

9.10 OnClosePic

关闭流程图事件。当关闭流程图后，函数被执行。

例如：在流程图关闭时，给出提示。编辑脚本如下：

```

Sub Form_OnClosePic()
    App.MsgBox "流程图将关闭", "SCADA", 0
End Sub

```

当脚本被执行后，流程图关闭时，将弹出标题栏为“SCADA”，内容为“流程图将关闭”的提示框。

9.11 OnOpenPic

打开流程图事件。当打开流程图后，函数被执行。

例如：在流程图打开时，给出提示。编辑脚本如下：

```

Sub Form_OnOpenPic()

```

```
App.MsgBox "正在打开流程图", "SCADA", 0  
End Sub
```

当脚本被执行后，流程图打开时，将弹出标题栏为“SCADA”，内容为“正在打开流程图”的提示框。

10 调度支持的脚本



提示：

关于调度组态的内容，请参见《组态管理软件使用手册》。

10.1 时间计划的脚本

时间计划脚本是指时间计划相关的脚本，一般以“TimerN”为句首。其中，“TimerN”为时间计划名称，默认的“TimerN”是系统为时间计划自动添加的名称。

时间计划的脚本分为函数和事件两种。函数主要用于对时间计划的属性进行查询或操作，事件是主要用于触发时间计划。

10.1.1 时间计划的接口函数

时间计划的接口函数包括 DayOfMonth、DayOfWeek 等。

DayOfMonth

DayOfMonth 函数用于显示每月时间计划中的日期信息。

- 语法
- Timer 名.DayOfMonth
- 输入参数
- 无
- 返回值
- 每月型时间计划中勾选的日期。
- 举例
- 在设置时间计划的对话框中将触发信息选择为每月并勾选星期天数，然后设置开始时间和停止时间使能(可选)。打开调度脚本编辑界面，输入以下语句，则软件会在设定的时间显示所勾选的每月天数
- msgbox timer1.DayOfMonth

DayOfWeek

DayOfWeek 函数用于显示每周时间计划中配置的星期信息。

- 语法
- Timer 名.DayOfWeek
- 输入参数
- 无
- 返回值
- 每周型时间计划中勾选的星期数。
- 举例

- 在设置时间计划的对话框中将触发信息选择为每周并勾选星期天数，然后设置开始时间和停止时间使能。打开调度脚本编辑界面，输入以下语句，则软件会在设定的时间显示所勾选的星期数。
- `msgbox timer1.DayOfWeek`

Description

Description 函数用于显示时间计划的描述信息。

- 语法
- `Timer 名.Description`
- 输入参数
- 无
- 返回值
- 时间计划的描述信息
- 举例
- 在设置时间计划的对话框中填写调度信息中的描述内容，并且选择触发信息并设置好开始时间与时间间隔。打开调度脚本编辑界面，输入以下语句，则软件会定时弹出消息框，显示所填写的调度信息的描述内容。
- `msgbox timer1.Description`

EnableEndTime

EnableEndTime 函数用于显示时间计划是否设置了停止时间。

- 语法
- `Timer 名.EnableEndTime`
- 输入参数
- 无
- 返回值
- 0 或 1，用于表示是否已设置了停止时间。0 表示未设置停止事件，1 表示已设置了停止时间。
- 举例
- 在提示框中显示 `timer1` 是否设置了停止时间。
- `msgbox timer1.EnableEndTime`

EndTime

EndTime 函数用于显示时间计划已设置的停止时间。

- 语法
- `Timer 名.EndTime`
- 输入参数
- 无
- 返回值
- 时间计划的停止时间。
- 举例

- 在设置时间计划的对话框中将触发信息选择为每周或每月并设置开始时间，勾选停止时间使能后设置停止时间与时间间隔。打开调度脚本编辑界面，输入以下语句，则软件会在设定的时间显示所设置的停止时间。
- `msgbox timer1.EndTime`

Interval

Interval 函数用于显示时间计划中设置的时间间隔。

- 语法
- `Timer 名.Interval`
- 输入参数
- 无
- 返回值
- 时间计划中设定的时间间隔。
- 举例
- 在设置时间计划的对话框中将触发信息选择为连续或每周或每月并设置开始时间与时间间隔。打开调度脚本编辑界面，输入以下语句，则软件会在设定的时间显示所设置的时间间隔。
- `msgbox timer1.Interval`

StartTime

StartTime 函数用于显示时间计划的开始时间。

- 语法
- `Timer 名.StartTime`
- 输入参数
- 无
- 返回值
- 时间计划的开始时间。
- 举例
- 在设置定时操作的对话框中将触发信息选择为连续，每周和每月并设置开始时间，设定好时间间隔，打开调度脚本编辑界面，输入上述语句，则软件会在设定的时间间隔定时显示所设置的开始时间。
- `msgbox timer1.starttime`

StartTimer

StartTimer 函数用于开启定时器。

- 语法
- `Timer 名.StartTimer`
- 输入参数
- 无
- 返回值
- 无

- 举例
- 在设置事件操作的对话框中选择子工程的位号，事件类型设置为为假时，打开调度脚本编辑界面，输入以下语句，则软件会在所选位号变为 OFF 时开启 timer1 定时器。
- Sub Event1_OnFalse()
 - timer1.starttimer
- End Sub

StopTimer

StopTimer 函数用于显示关闭定时器。

- 语法
- Timer 名.StopTimer
- 输入参数
- 无
- 返回值
- 无
- 举例
- 在设置事件操作的对话框中选择子工程的位号，事件类型设置为为真时。打开调度脚本编辑界面，输入以下语句，则软件会在所选位号变为 ON 时停止定时器功能。
- Sub Event1_OnTrue()
 - timer1.stoptimer
- End Sub

TimerEnabled

TimerEnabled 函数用于返回定时器的状态。

- 语法
- Timer 名.TimerEnabled
- 输入参数
- 无
- 返回值
- 0 或 1，0 表示定时器已启动，1 表示定时器未启动。
- 举例
- 返回定时器 timer1 的状态。
- msgbox timer1.TimerEnabled

TriggerType

TriggerType 函数用于显示时间计划的触发信息类型。

- 语法
- Timer 名.TriggerType
- 输入参数
- 无
- 返回值

- 0~3，用于表示定时操作的触发信息类型。0 表示单次；1 表示连续；2 表示每周；3 表示每月。
- 举例
- 在设置时间计划的对话框中首先选择触发信息并设置好开始时间与时间间隔，打开调度脚本编辑界面，输入上述语句，则软件会定时弹出消息框，显示所选择的触发信息类型。
- `msgbox timer1.TriggerType`

10.1.2 时间计划的事件函数 OnTimeOut

OnTimeOut 函数用于定义时间到后触发函数内的脚本。

- 语法
- `Sub 时间计划名_OnTimeOut(strTimerName)`
- 待触发的内容
- `End Sub`
- 举例
- 当时间计划“Timer1”的时间到后，弹出提示框“OK”。
- `Sub Timer1_OnTimeOut(strTimerName)`
- `msgbox "ok" '时间到了，弹出消息框 OK`
- `End Sub`

10.2 事件计划相关脚本

事件计划脚本是指事件操作相关的脚本，一般以“EventN”为句首，其中“EventN”为事件名，默认的“EventN”是系统为定时操作自动添加的名称。

事件计划的脚本分为函数和事件两种。函数主要用于对事件计划的属性进行查询或操作，事件是主要用于触发事件计划。

10.2.1 事件计划的接口函数

事件计划的接口函数包括 Description、EventEnabled、EventType 等。

Description

Description 函数用于获取指定事件计划的描述信息。

- 语法
- `事件计划名.Description`
- 输入参数
- 无
- 返回值
- 事件计划的描述信息。
- 举例
- 在“Event1”设置事件操作的对话框中填写计划信息中的描述项，打开调度脚本编辑界面，输入以下语句，则软件会在设定的时间显示所填写的事件计划信息。
- `msgbox Event1.Description`

EventEnabled

EventEnabled 函数用于获取指定事件计划的状态。

- 语法
- 事件计划名.EventEnabled
- 输入参数
- 无
- 返回值
- 事件计划的状态，0 或 1。0 表示事件计划未开启，1 表示事件计划已开启。
- 举例
- 在调度中添加事件计划“Event1”，并输入以下脚本。在脚本触发时，会弹出“Event1”的状态信息。
- msgbox Event1.EventEnabled

EventType

EventType 函数用于获取指定事件计划的类型。

- 语法
- 事件计划名.EventType
- 输入参数
- 无
- 返回值
- 事件计划的类型，0~4。0：变化时、1：为真时、2：为假时、3：总为真、4：总为假。
- 举例
- 在调度中添加事件计划“Event1”，并输入以下脚本。在脚本触发时，会弹出“Event1”的类型。
- msgbox Event1.EventType

Expression

Expression 函数用于获取指定事件计划的表达式。

- 语法
- 事件计划名.Expression
- 输入参数
- 无
- 返回值
- 事件计划的表达式。
- 举例
- 在调度中添加事件计划“Event1”，并输入以下脚本。在脚本触发时，会弹出“Event1”的表达式。
- msgbox Event1.Expression

Interval

Interval 函数用于获取指定事件计划的时间间隔。

- 语法
- 事件计划名.Interval
- 输入参数
- 无
- 返回值
- 事件计划的时间间隔。
- 举例
- 在调度中添加事件计划“Event1”，并输入以下脚本。在脚本触发时，会弹出“Event1”的时间间隔。
- msgbox Event1.Interval

StartEvent

StartEvent 函数用于启动指定的事件计划。

- 语法
- 事件计划名.StartEvent
- 输入参数
- 无
- 返回值
- 无
- 举例
- 在调度中添加事件计划“Event1”，并输入以下脚本。在脚本触发时，会启动“Event1”。
- msgbox Event1.StartEvent

StopEvent

StopEvent 函数用于停止指定的事件计划。

- 语法
- 事件计划名.StopEvent
- 输入参数
- 无
- 返回值
- 无
- 举例
- 在调度中添加事件计划“Event1”，并输入以下脚本。在脚本触发时，停止“Event1”。
- msgbox Event1.StopEvent

10.2.2 事件计划的事件函数

事件计划的事件函数包括 OnTrue、OnFalse 等。

OnTrue

OnTrue 函数用于定义事件状态中定义的位号为真时，触发的事件。

- 语法
- Sub 事件计划名_ OnTrue()
- 待触发的内容
- End Sub
- 举例
- 事件 “Event1” 中关联位号为真时，弹出消息框"event1_ontrue"。
- Sub Event1_ OnTrue()
- msgbox "event1_ontrue"
- End Sub

OnFalse

OnFalse 函数用于定义事件状态中定义的位号为假时，触发的事件。

- 语法
- Sub 事件计划名_ OnFalse ()
- 待触发的内容
- End Sub
- 举例
- 事件 “Event1” 中关联位号为假时，弹出消息框"event1_ OnFalse"。
- Sub Event1_ OnFalse ()
- msgbox "event1_ OnFalse"
- End Sub

WhileTrue

WhileTrue 函数用于定义事件状态中关联位号值由假变为真时触发的事件，以及位号值一直为真时每隔一定周期触发的事件。

- 语法
- Sub 事件计划名_ WhileTrue ()
- 待触发的内容
- End Sub
- 举例
- 事件 “Event1” 中关联位号值由假跳变为真时，弹出消息框"event1_ WhileTrue"。
- Sub Event1_ WhileTrue ()
- msgbox "event1_ WhileTrue"
- End Sub

WhileFalse

WhileFalse 函数用于定义事件状态中关联位号的值由真变为假时触发的事件，以及位号值一直为假时每隔一定周期触发的事件。

- 语法
- Sub 事件计划名_ WhileFalse ()
- 待触发的内容
- End Sub

- 举例
- 事件“Event1”中关联的位号值由真跳变为假时，弹出消息框"event1_ WhileFalse"。
- Sub Event1_ WhileFalse ()
- msgbox "event1_ WhileFalse"
- End Sub

DataChange

DataChange 函数用于定义事件状态中配置的位号值变化时，触发的事件。

- 语法
- Sub 事件计划名_ DataChange (fCurValue)
- 待触发的内容
- End Sub
- 举例
- 事件“Event1”中关联的位号值变化时，弹出消息框"event1_ WhileFalse"。
- Sub Event1_ DataChange ()
- msgbox "event1_ WhileFalse"
- End Sub

OnKeyPress

OnKeyPress 函数用于定义当键盘按键按下或弹起时，触发的事件。

- 语法
- Sub 事件计划名_OnKeyPress()
- 待触发的内容
- End Sub
- 举例
- 当某按键按下或弹起时，弹出消息框“当前操作小组是 XXX”。
- Sub Event1_OnKeyPress()
- group = App.GetGroup
- App.MsgBox "当前操作小组是" & group, "SCADA", 0
- End Sub

10.3 调度支持的事件

调度支持的事件是指脚本中“Scheduler”相关的事件函数，包括“Initialize”和“Close”。

10.3.1 Initialize

Initialize 函数用于定义系统初始化时，触发的事件。

- 语法
- Sub Scheduler_ Initialize()
- 待触发的内容
- End Sub
- 举例

- 当系统初始化开始时，弹出信息框显示“调度初始化事件开始”。
- Sub Scheduler_Initialize()
- msgbox("调度初始化事件开始")
- End Sub

10.3.2 Close

Close 函数用于定义系统退出时，触发的事件。

- 语法
- Sub Scheduler_Close ()
- 待触发的内容
- End Sub
- 举例
- 当系统退出时，弹出信息框显示“系统正在退出”。
- Sub Scheduler_Close ()
- msgbox("系统正在退出")
- End Sub

10.4 调度支持的 APP 函数

10.4.1 OnSwitchServer

OnSwitchServer 函数定义当主从服务器发生切换时，触发的事件。

语法

```
Sub App_OnSwitchServer ()
    待触发的内容
End Sub
```

举例

当切换主从服务器时，D 盘 Logfile 文件中将显示主服务器 IP 地址。

```
Sub App_OnSwitchServer()
    Set a = App.ServerInfo
    Set b = a.GetPrimaryServerIP '查找主服务器 IP
    b.MoveFirst '循环语句，若主服务器网络冗余，则将显示两个 IP
    do Until b.EOF
        ip1 = ip1 + " " + b.IPValue
        b.MoveNext
    loop
    VxDebug.LogFile "主:"&ip1, "D:\Logfile.txt"
End Sub
```

10.4.2 OnSwitchTeam

OnSwitchTeam 函数定义当操作小组发生切换时，触发的事件。

语法

```
Sub App_OnSwitchTeam(oldTeam, newTeam)
    待触发的内容
End Sub
```

举例

当切换操作小组时，弹出提示信息“当前操作小组是 XXXX”。

```
Sub App_OnSwitchTeam(oldTeam, newTeam)
    group = App.GetGroup
    App.MsgBox "当前操作小组是" & group, "SCADA", 0
End Sub
```

10.4.3 OnUserLogin

OnUserLogin 函数定义监控用户登录或注销时，触发的事件。

语法

```
Sub App_OnUserLogin(username, login)
    待触发的内容
End Sub
```

举例

当监控用户 admin 登录时，将弹出提示信息“欢迎使用 InPlant SCADA”。

```
Sub App_OnUserLogin(username, login)
    If username="Admin" and login=1 then
        App.MsgBox "欢迎使用 InPlant SCADA ", "InPlant SCADA", 0
    End if
End Sub
```

注意

用户注销时，login=0

10.5 注意事项

1. 全局调度和操作小组调度不建议弹出提示框（MsgBox 或 App.MsgBox）。
2. 如果必须弹框，建议放在独立的调度中，以免影响其他调度的正常执行。
3. 全局调度脚本中，若对结构位号进行写值，则需进行类型强转操作，如：
`tag("A.B") = CDBL(tag("A.B")) + 1`

11 脚本应用举例

11.1 从第三方数据库读数据

11.1.1 背景

读取第三方数据库油田数据库中的数据后，通过示功图的形式表达出来。

准备

- 数据库：MySQL V5.1
- 数据库地址：172.20.1.50
- 本地安装数据库驱动程序：MySQL ODBC 5.1 Driver，并进行数据源（ODBC）的设置。
以 Windows 7 系统为例，打开【控制面板/系统和安全/管理工具/数据源（ODBC）】，在“用户 DSN”页面中点击“添加”按钮，弹出如图 11-1 所示界面，选择数据源 MySQL ODBC 5.1 Driver，点击“完成”。



图 11-1 创建新数据源

上述界面选择并点击完成后，弹出如图 11-2 所示界面，设置数据库地址、用户名、密码和数据库名称。

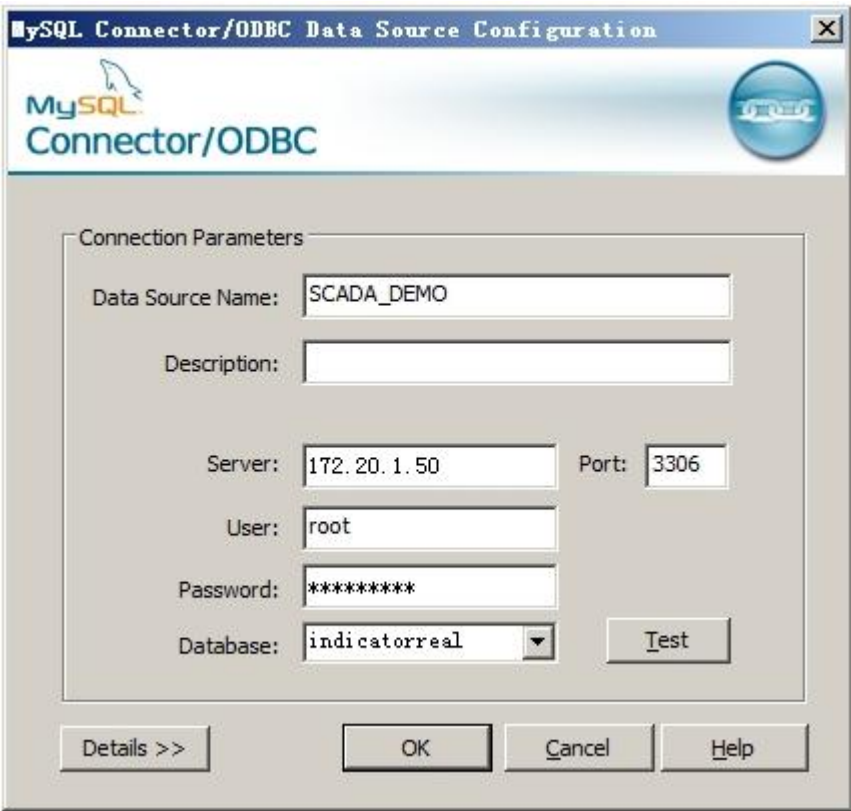


图 11-2 ODBC 设置界面

- 数据库用户名和密码：root /supcondcs
- 端口：3306
- 控件：功图控件（ICHART）

读取的数据库信息

Field	Type	Null	Key	Default	Extra
ID	int(4)	NO	UNI	NULL	auto_increment
F_START_TIME	datetime	NO	PRI	NULL	
F_TIME_MILI	int(4)	YES		NULL	
F_NUM	int(4)	YES		NULL	
F_DATA	blob	NO		NULL	
F_EXTDATA	blob	YES		NULL	
F_INDEX	int(4)	YES		NULL	
F_CSNAME	varchar(64)	NO	PRI		
F_FEAFLAG	int(4)	YES		NULL	
F_DEVID	int(4)	NO		NULL	
F_CIRCLE	int(4)	NO		NULL	
F_STOKE	int(4)	NO		NULL	

图 11-3 读取的数据库信息

11.1.2 代码

```
Function ConnectDB
'创建数据库对象
Set recordConnection = CreateObject("ADODB.Connection")
```

Dim connectString

'数据库连接信息的字符串，包括数据库类型，服务器地址，用户名，密码等

```
connectString = "Provider=MSDASQL.1;Persist Security Info = True; Extended
Properties='driver={MySQL ODBC 5.1 Driver}; server = 172.20.1.50;
UID=root;PWD=supcondcs;DATABASE=indicatorreal;PORT=3306"
```

'创建数据库连接

```
recordConnection.open connectString
```

'数据库连接成功，启动定时器

```
If recordConnection.State = 1 Then
```

```
    Timer1.Enable = True
```

```
End If
```

```
End Function
```

'定时器,读取最新数据

```
Sub Timer1_OnTimer()
```

```
    strQuery = "select * from indicatorreal.main order by F_START_TIME desc limit 1" ' 读
indicatorreal.main 表中以 F_START_TIME 降序排列的第一个数据
```

'创建数据库 recordset 对象，用于数据查询

```
Set recordSet = CreateObject("ADODB.Recordset")
```

'创建 recordset 连接

```
recordSet.open strQuery, recordConnection
```

'清除图表数据

```
IChart1.ClearAllSeries '清空示功图
```

```
While Not recordSet.EOF '遍历
```

'查询数据

```
ui1 = recordSet("DATA").value '读取 DATA 中的数据赋值给 ui1
```

```
ui2 = recordSet("NUM").value '读取 NUM 中的数据赋值给 ui2
```

IChart1.GetInspectionData ui2, ui1, 0, 0 '获取油田数据库中的某个数据点，ui2 是数组数量，ui1 是数据库中的数据，载荷数据，偏移 0

```
t = 0
```

```
While t < ui2
```

```
    x1 = IChart1.GetSpecialData (0, t) '控件获取特定字段特定值，载荷数据，偏移 t
```

```
    y1 = IChart1.GetSpecialData (1, t) '获取特定字段特定值，位移数据，偏移 t
```

'在图表上添加一个点

```
IChart1.AddPointSingle 0, x1, y1 '添加一个点，在原有曲线上添加点，X 轴 Y 轴对应值分别为
x1, y1
```

```
    t = t + 1
```

```
WEND
```

```
IChart1.SetSeparated(True) '上下冲程分色显示
```

IChart1.Refresh '刷新功图控件

'移到下一条

recordSet.MoveNext

WEND

'关闭 recordset，然后置成空

recordSet.Close

Set recordSet = Nothing

End Sub

11.1.3 结果

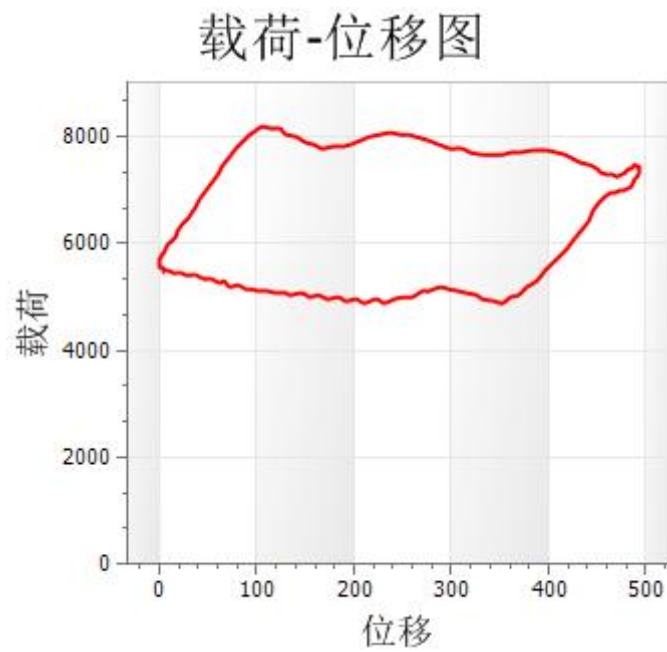


图 11-4 载荷-位移图

11.2 写数据到第三方数据库

11.2.1 背景

将某段时间内报警的所有位号的报警状态和时间写入第三方数据库。

准备

- 数据库：MySQL V5.1
- 数据库地址：127.0.0.1
- 本地安装数据库驱动程序：MySQL ODBC 5.1 Driver，并进行数据源（ODBC）的设置。
以 Windows 7 系统为例，打开【控制面板/系统和安全/管理工具/数据源（ODBC）】，在“用户 DSN”页面中点击“添加”按钮，弹出如图 11-5 所示界面，选择数据源 MySQL ODBC 5.1 Driver，点击“完成”。



图 11-5 创建新数据源

上述界面选择并点击完成后，弹出如图 11-6 所示界面，设置数据库地址、用户名、密码和数据库名称。



图 11-6 ODBC 设置界面

- 数据库用户名和密码: root /supcondcs
- 端口: 3306

11.2.2 代码

'创建数据库对象

```
Set recordConnection = CreateObject("ADODB.Connection")
```

```
Dim connectString
```

'数据库连接的信息

```
connectString = "Provider=MSDASQL.1;Persist Security Info = True; Extended  
Properties='driver={MySQL ODBC 5.1 Driver}; server = 127.0.0.1;  
UID=root;PWD=supcondcs;DATABASE=indicatorreal;PORT=3306'"
```

'创建数据库连接的信息

```
recordConnection.open connectString '连接数据库
```

```
Set cat = CreateObject("ADOX.Catalog") '访问数据库的管理对象
```

```
Set table = CreateObject("ADOX.Table") '访问表的管理对象
```

'创建表

```
recordConnection.execute "CREATE TABLE if not exists `RealData` (`TagName` VARCHAR(256) NOT  
NULL, `AlarmStatus` VARCHAR(256), `AlarmTime` DATETIME)"
```

'读位号历史报警信息，并将某些位号的报警信息写入数据库表中

```
Set a = App.Alarm
```

```

Set b = a.HistoryAlarm
Set c = b.QueryAlarm ("2014-12-5 09:30:00", "2014-12-5 09:40:00", 100, 0)
c.MoveFirst
Do Until c.EOF
    recordConnection.execute "INSERT INTO RealData ( TagName , AlarmStatus , AlarmTime )
VALUES('&c.tagName&', '&c.Status&', '&c.Time&')" '将位号值写入数据库表中
    c.MoveNext
Loop

```

11.2.3 结果

TagName	AlarmStatus	AlarmTime
▶ TAG0	1	2014-12-16 08:00:00
TAG1	0	2014-12-16 09:00:00
TAG2	1	2014-12-16 10:00:00
TAG3	0	2014-12-16 11:00:00
TAG4	0	2014-12-16 12:00:00

图 11-7 写完后数据库中的数据



11.3 全局脚本应用举例

11.3.1 背景

全局脚本，用于在监控启动时，对系统进行初始化，在监控退出时进行一些资源的释放，在系统运行期间用户和操作小组切换时可触发指定的脚本等操作。

全局脚本的基本操作步骤如下所述。

11.3.2 操作步骤

1. 打开组态管理软件，子工程下的“调度”，通过按钮添加调度文件，并打开调度组态界面。具体操作方式可参照《组态管理软件使用手册》中“调度”章节内容。
2. 在调度组态软件中，点击工具栏图标，打开调度脚本编辑器，在编辑器中添加 Scheduler 和 App 的全局脚本，如下图所示。

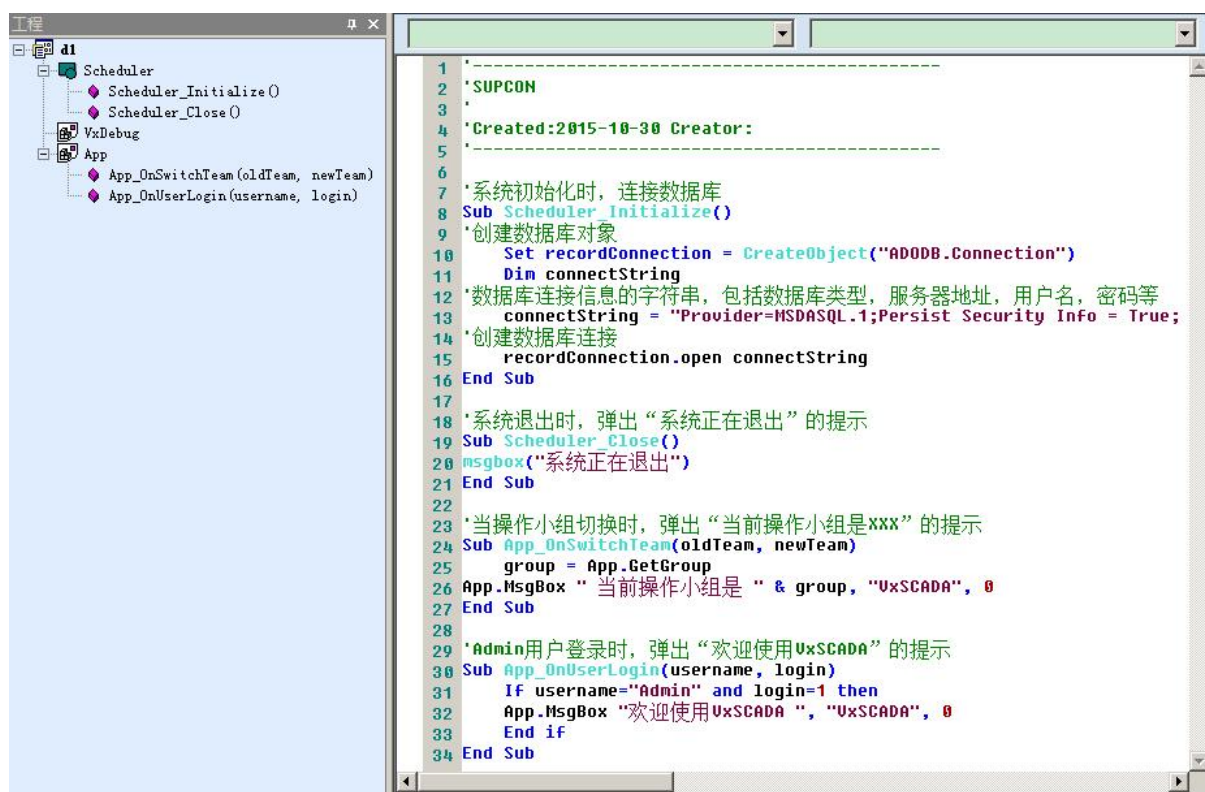


图 11-8 全局脚本应用实例

3. 可在不同的调度中根据工程的需要在上述全局事件中添加脚本,此处的脚本在整个系统运行期间均有效。

11.3.3 结果

脚本发布,并在系统监控系统启动后,在相应事件被触发后将执行事件内定义的脚本。

12 资料版本说明

表 12-1 版本升级更改一览表

资料版本号	适用软件版本	更改说明
V2.1（20221220）	InPlant SCADA V5.50.00.00 及以上版本	勘误
V2.2（20230411）	InPlant SCADA V5.50.01.00 及以上版本	新增数据库表控件专用接口
V2.3（20230629）	InPlant SCADA V5.50.02.00 及以上版本	脚本应用举例勘误
V2.4（20231113）	InPlant SCADA V5.50.03.00 及以上版本	更新数据库表控件图标

13 附录

13.1 颜色常数

这些常数是在 VFScript 中设置的，使用之前不必定义它们，可在代码中随时使用。

表 13-1 颜色常数

常数	值	描述
vbBlack	&h00	黑色
vbRed	&hFF	红色
vbGreen	&hFF00	绿色
vbYellow	&hFFFF	黄色
vbBlue	&hFF0000	蓝色
vbMagenta	&hFF00FF	紫色
vbCyan	&hFFFF00	青色
vbWhite	&hFFFFFF	白色

13.2 背景风格对照表

表 13-2 背景风格对照表

风格名称	值
实心填充	0 - bgSOLIDFILL
不填充	1 - bgNOHATCH
左斜线	5 - bgBDIAGONAL
右斜线	4 - bgFDIAGONAL
十字网格	6 - bgCROSS
对角网格	7 - bgDIAGCROSS
垂直线	3 - bgVERTICAL
水平线	2 - bgHORIZONTAL

13.3 边框风格对照表

表 13-3 边框风格对照表

风格名称	值
无	5-esNOLINE
实线	0-esSOLID
虚线	1-esDASH
点线	2-esDOT
点划线	3-esDASHDOT
双点划线	4-esDASHDOTDOT


13.4 渐变风格对照表

表 13-4 渐变风格对照表

风格名称	值
实心填充	0 - gsSOLID
由左至右变	1 - gsLEFTRIGHT
由上至下变	2 - gsTOPBOTTOM
由中间至左右两边变	3 - gsHCENTER
由中间至上下两边变	4 - gsVCENTER
左上放射渐变	5 - gsTOPLEFT
左下放射渐变	6 - gsBOTTOMLEFT
右下放射渐变	7 - gsBOTTOMRIGHT
右上放射渐变	8 - gsTOPRIGHT
左下渐变	9 - gsLEANDOWN
左上渐变	10 - gsLEANUP
由中心向四周变	11 - gsCENTER

13.5 按钮风格对照表

表 13-5 按钮风格对照表

风格样式	值
	0-esRECTANGLE
	1-esFLAT
	2-esELLIPSE
	3-esNOISE
	4-esDIAGSHADE
	5-esHSHADE（默认值）
	6-esVSHADE
	7-esHBUMP
	8-esVBUMP
	9-esSOFTBUMP
	10-esHARDBUMP
	11-esMETAL

13.6 MsgBox 中按钮设置和返回值说明

表 13-6 Button 设置值与样式对应关系

值	按钮
0	只显示“确定”按钮
1	显示“确定”和“取消”按钮
2	显示“放弃”、“重试”，“忽略”按钮
3	显示“是”、“否”和“取消”按钮
4	显示“是”、“否”按钮
5	显示“重试”和“取消”按钮

表 13-7 按钮与对应返回值

值	按钮
---	----

值	按钮
1	确定
2	取消
3	放弃
4	重试
5	忽略
6	是
7	否

13.7 报警统计条件字符串

表 13-8 报警统计 RTAC 条件字符串

过滤器表达	语义	示例
PRI(x, y)	优先级在[x, y]全闭区间，未设置表示不过滤优先级	PRI(0,31) 优先级 0 到 31 PRI(2,2) 优先级 2
ACK(x)	报警是否确认，未设置表示不过滤确认状态	ACK(1) 已确认报警 ACK(0) 未确认报警
TYPE(a, b, c, ...)	过滤报警类型，参数不定数量，最多六个，可选范围为 HH 过滤高高限报警，H 过滤高限报警，L 过滤低限报警，LL 过滤低低限报警，ON 过滤开报警，OFF 过滤关报警，未设置表示不过滤报警类型	TYPE(H,L,ON,OFF)表示过滤高限，低限，开和关报警。
GROUP()	过滤报警分组分区，未设置表示不过滤报警分组分区	GROUP(G0,G1,G2(0,1))表示过滤报警 0 组、1 组、2 组 0 区、2 组 1 区
ALMGROUP()	过滤报警分组分区，未设置表示不过滤报警分组分区	ALMGROUP(G0,G1,G2(0,1))表示过滤报警 0 组、1 组、2 组 0 区、2 组 1 区
SLEEP(x)	过滤瞌睡报警，未设置表示不过滤瞌睡报警	SLEEP(0) 正常报警 SLEEP(1) 瞌睡报警

表 13-9 报警统计 RTAlmStat 条件字符串

过滤器表达	语义	示例
PRI(x, y)	优先级在[x, y]全闭区间，未设置表示不过滤优先级	PRI(0,31) 优先级 0 到 31 PRI(2,2) 优先级 2
ACK(x)	报警是否确认，未设置表示不过滤确认状态	ACK(1) 已确认报警 ACK(0) 未确认报警
TYPE(a, b, c, ...)	过滤报警类型，参数不定数量，最多六个，可选范围为 HH 过滤高高限报警，H 过滤高限报警，L 过滤低限报警，LL 过滤低低限报警，ON 过滤开报警，OFF 过滤关报警，未设置表示不过滤报警类型	TYPE(H,L,ON,OFF)表示过滤高限，低限，开和关报警。

过滤器表达	语义	示例
PRIGROUP(分组名)	过滤优先级分组名, 未设置表示不过滤优先级分组名	PRIGROUP(小组 1)表示过滤优先级分组为小组 1 的报警信息
GROUP()	过滤报警分区分区, 未设置表示不过滤报警分区分区	GROUP(G0,G1,G2(0,1))表示过滤报警 0 组、1 组、2 组 0 区、2 组 1 区
ALMGROUP()	过滤报警分区分区, 未设置表示不过滤报警分区分区	ALMGROUP(G0,G1,G2(0,1))表示过滤报警 0 组、1 组、2 组 0 区、2 组 1 区
SLEEP(x)	过滤瞌睡报警, 未设置表示不过滤瞌睡报警	SLEEP(0) 正常报警 SLEEP(1) 瞌睡报警
OPERATOR(op)	过滤报警确认人, 未设置表示不过滤报警确认人	OPERATOR(admin)表示过滤报警确认为 admin 的报警信息

表 13-10 报警统计 RTAlmStat 返回值的属性和方法

类别	语义	示例
属性	Count, 返回当前集合的报警数目	RTALMSTAT("PRI(0,10)").COUNT
方法	<ol style="list-style-type: none"> Max("subsubFilter"), 返回集合的 subsubFilter 条件字段中最大的数 (字符串格式)。 Min("subsubFilter"), 返回集合的 subsubFilter 条件字段中最小的数 (字符串格式)。 subsubFilter 包含以下几种: <ul style="list-style-type: none"> AlmPri, 报警优先级 AlmGroupID, 报警分组 ID AlmSubAreaID, 报警分区 ID AlmTime, 报警产生时间 AlmAckTime, 报警确认时间 AlmRemoveTime, 报警移除时间 	<pre>Set a = App.Alarm Set b = a.RTAlarm b.SetRTAlmFilterEx("RTALMSTAT("PRI(0,10)").Max("AlmTime")") result = b.GetRTStatResult() app.MsgBox CStr(result), "MaxAlmTime", 0</pre>

13.8 报警类型与子条件对照表

表 13-11 各类报警对应的子条件

数据类型	报警类型		子条件 (dwSubCondition)		报警类型 (almType)
			十进制	十六进制	
实型、整型	低三限报警		65536	0x10000	L3
	低低限报警		2048	0x0800	LL
	低限报警		512	0x0200	L
	高限报警		256	0x0100	H
	高高限报警		1024	0x0400	HH
	高三限报警		131072	0x20000	H3
	超上限		16384	0x4000	ORH
	超下限		32768	0x8000	ORL
	变化率报警	正变化率报警	4096	0x1000	PRIN

数据类型	报警类型		子条件（dwSubCondition）		报警类型（almType）
		负变化率报警	8192	0x2000	NRIN
开关量	ON 报警		256	0x0100	ON
	OFF 报警		512	0x0200	OFF