

CORWARE 软件使用手册

V 3.2.3

正知（上海）智能技术有限公司

目录

第一章 关于 CORWARE 的安装-----	6
1.1 安装 CORWARE-----	6
软件来源: -----	6
电脑硬件要求: -----	6
作业系统:-----	6
1.2 安装步骤-----	7
第二章 简单使用 CORWARE-----	11
2.1 新建工程-----	11
2.2 新建 I/O 变量: -----	13
2.3 创建组态画面-----	19
2.4 建立动画连接-----	20
2.5 离线模拟-----	23
2.6 下载工程到 HMI-----	24
第三章 CORWARE 菜单栏详解-----	26
3.1 工程菜单-----	26
3.2 编辑菜单-----	26
3.3 查看菜单-----	28
3.4 插入菜单-----	30
3.5 布局菜单-----	错误! 未定义书签。
3.6 工具菜单-----	50
3.7 帮助菜单-----	55
第四章 工程管理器-----	56
4.1 工程管理器概述-----	57
4.2 画面-----	57
启动画面-----	58
主画面-----	58
正常画面-----	58
4.3 窗口-----	58
4.4 数据库-----	60
4.5 全局脚本-----	60
4.6 配方管理-----	61
4.7 用户管理-----	63
4.8 报表。-----	64
第五章 构造数据库-----	65
5.1 IO 设备管理-----	65
5.2 定义变量-----	65
5.3 变量类型-----	70
5.4 系统变量-----	71
5.5 变量的数据类型-----	71
第六章 画面动态连接-----	73
6.1 动态连接属性框-----	74

6.2 动态连接表达式-----	75
6.2.1 ^ 运算符-----	75
6.2.2 * 运算符-----	75
6.2.3 / 运算符-----	75
6.2.4 + 运算符-----	76
6.2.5 - 运算符-----	76
6.3 动态连接详解-----	76
6.3.1 线条颜色-----	76
6.3.2 填充颜色-----	77
6.3.3 文字颜色-----	77
6.3.4 水平移动-----	78
6.3.5 垂直移动-----	79
6.3.6 宽度变化-----	80
6.3.7 高度变化-----	80
6.3.8 数值显示-----	81
6.3.9 文本显示-----	82
6.3.10 图符显示-----	83
6.3.11 可见性-----	83
6.3.12 闪烁-----	84
6.3.13 设置数值-----	84
6.3.14 设置状态-----	85
6.3.15 切换画面-----	86
6.3.16 弹出窗口-----	86
6.3.17 加载配方-----	87
第七章 曲线显示-----	89
7.1 实时曲线-----	90
7.1.1 创建实时曲线图元-----	90
7.1.2 实时曲线属性设置-----	90
7.2 历史曲线-----	93
7.2.1 创建历史曲线图元-----	93
7.2.2 历史曲线属性设置-----	94
7.3 计划曲线-----	95
7.3.1 创建计划曲线图元-----	96
7.3.2 计划曲线属性设置-----	96
7.4 XY 曲线-----	97
第八章 报表系统-----	100
8.1 如何创建报表-----	102
8.1.1 新建报表-----	102
8.1.2 报表单元格样式-----	103
8.2 加载和查询报表-----	103
8.2.1 加载报表-----	104
8.2.2 查询报表-----	104
第九章 配方管理-----	105
9.1 概述-----	105

9.1.1 什么是配方-----	105
9.1.2 配方管理-----	106
9.1.3 如何使用配方-----	107
第十章 系统安全管理-----	108
10.1 开发系统安全管理-----	108
10.1.1 如何对工程进行加密-----	108
10.1.2 如何去除工程加密-----	110
10.1.3 如何设置工程上载密码-----	110
10.2 运行系统安全管理-----	111
10.2.1 运行系统安全管理概述-----	111
10.2.2 安全管理配置-----	111
10.2.3 运行时如何登录用户-----	112
10.2.4 与安全管理相关的系统变量和函数-----	112
第十一章 报警和事件系统-----	113
11.1 关于报警和事件-----	113
11.2 如何定义变量的报警属性-----	114
11.2.1 模拟量变量的报警类型-----	114
11.2.2 开关量变量的报警类型-----	118
11.3 事件类型及使用方法-----	118
11.3.1 操作事件-----	119
11.3.2 用户登录事件-----	120
11.4 报警图元-----	120
11.5 语音报警-----	121
第十二章 脚本语言-----	121
12.1 图元脚本-----	122
12.2 画面脚本-----	122
12.3 全局脚本-----	122
12.3.1 定时执行-----	123
12.3.2 触发执行-----	123
12.4 VBScript-----	123
12.4.1 VBScript 变量-----	123
12.4.2 条件语句-----	126
12.4.3 循环语句-----	128
12.4.4 实例-----	132
12.4.5 关键字-----	135
12.5 脚本函数-----	136
12.5.1 系统函数-----	136
12.5.2 时间日期函数-----	175
12.5.3 类型转换函数-----	183
12.5.4 格式化函数-----	187
12.5.5 数学函数-----	190
12.5.6 字符串函数-----	194
12.5.6 其他函数-----	200
第十三章 其他常用功能-----	205

13.1 工程下载方式-----	205
13.2 时间显示和时间修改-----	208
1. 时间显示-----	208
2. 时间修改-----	210
13.3 多语言-----	210
13.4 掉电保存-----	211
1. 方式一：脚本保存-----	211
2. 方式二：配方保存-----	212
13.5 设备变量导入和导出-----	212
1. 设备变量导出-----	212
2. 设备变量导入-----	213
13.6 Modbus 变量转发-----	214
1. 主机设置-----	214
2. 从机设置-----	216
13.7 背光控制-----	218
13.8 批量赋值-----	219
13.9 通讯控制-----	219
13.10 MAC 地址读取-----	220
13.11 码值-----	223
13.12 数据转换-----	223
13.13 音视频调用-----	225
13.13.1 音频-----	225
13.13.2 视频-----	225
13.14 图符库-----	226
13.15 历史文件导出-----	228
第十四章 工程测试-----	234
14.1 组态检查-----	234
14.2 外部设备的测试-----	235
14.3 动画动作的测试-----	235
14.4 按钮动作的测试-----	235
14.5 图形界面的测试-----	236
14.6 运行脚本的测试-----	236

第一章 关于 CORWARE 的安装

1.1 安装 CORWARE

软件来源:

1. 进入正知（上海）智能技术有限公司网站 www.cortek.cn 下载

下载链接: <http://www.cortek.cn/ztrj/>

2. 拨打服务热线: 400-0212-016

电脑硬件要求:

CPU: INTER Pentium II 以上等级

DDR: 256MB 以上

光驱: 4 倍速以上光驱一个

显示器: 支持解析度 1024 x 768 以上的彩色显示器

键盘和滑鼠各一个

以太网接口: [工程下载] / [工程上传] 时使用

USB 2.0: [工程下载] / [工程上传] 时使用

RS-232 COM 口: 线上模拟时使用

作业系统:

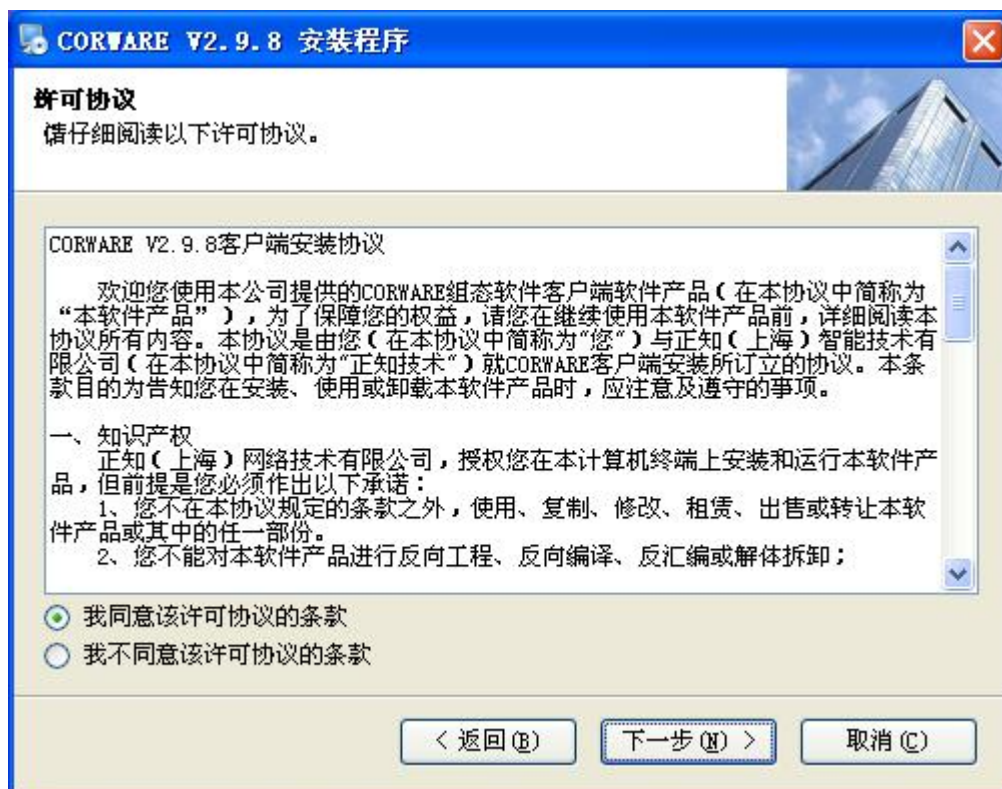
Windows XP / Windows 7/ Windows 10 均可。

1.2 安装步骤

1.2.1 下载安装程序后，双击 ，荧幕将显示安装程式如下：



1.2.2 许可协议条款确认：



1.2.3 指定一个全新的资料夹给 CORWARE 安装文件，或是直接使

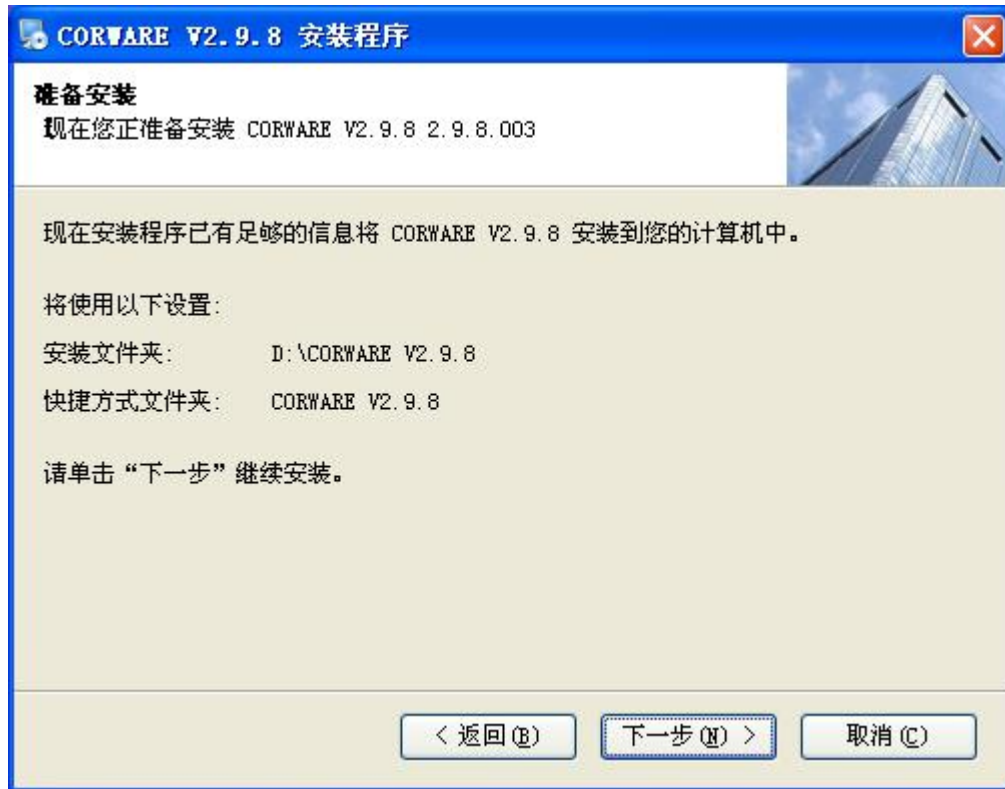
用系统建议的文件夹，然后点选 [下一步]。



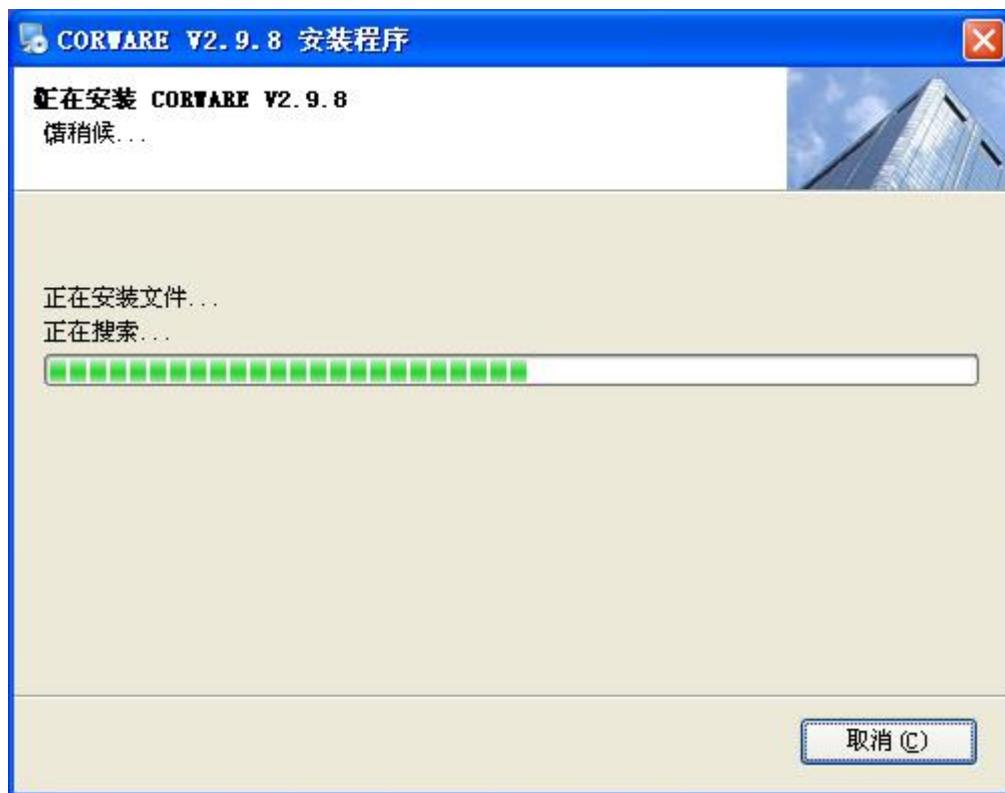
1.2.4 点选下一步



1.2.5 安装信息确认，点选“下一步”



1.2.6 安装程序执行中



1.2.7 安装完成



1.2.8 安装完成后会在桌面上生成软件快捷方式图标，在 [开始] / [所有程序] / [CORWARE 工业组态] 目录下可看到 CORWARE 组态软件快捷方式。




第二章 简单使用 CORWARE

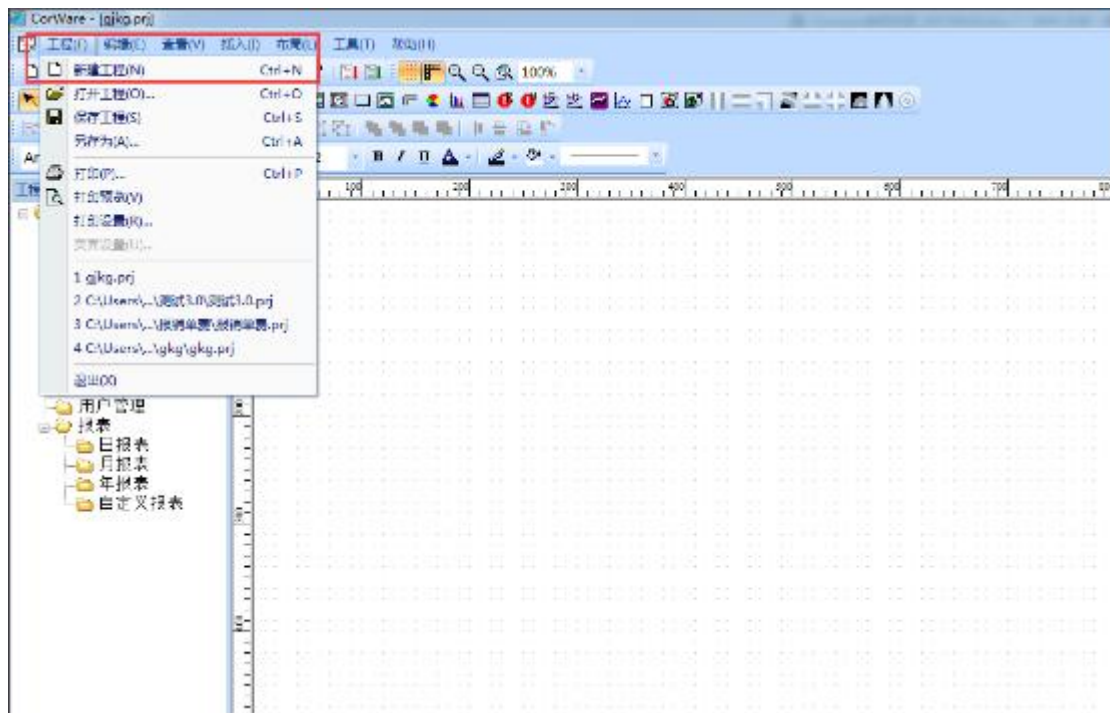
CORWARE 组态软件使用的一般步骤：

新建工程→建 I/O 变量→制作组态画面→关联变量→离线模拟→下载工程到 HMI

注：如果新建的工程有配方管理，则同时也需要点击工具中的下载配方，否则，下载的工程中没有配方项目。

2.1 新建工程

2.1.1 打开 CORWARE 组态软件，点击【工程】【新建工程】，或直接点击 ，或使用快捷键 Ctrl+N



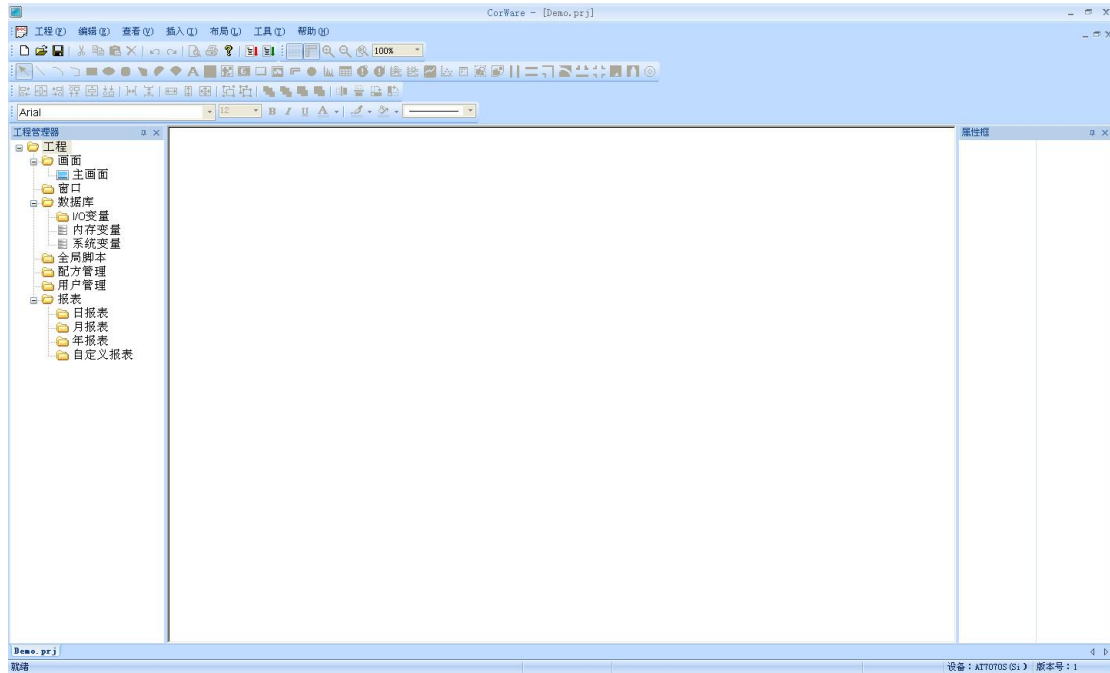
2.1.2 在弹出的窗口中输入工程名称和工程保存路径，点下一步



2.1.3 选择对应的 HMI 型号，点击【完成】



2.1.4 新建完成后即进入工程编辑界面。



2.2 新建 I/O 变量:

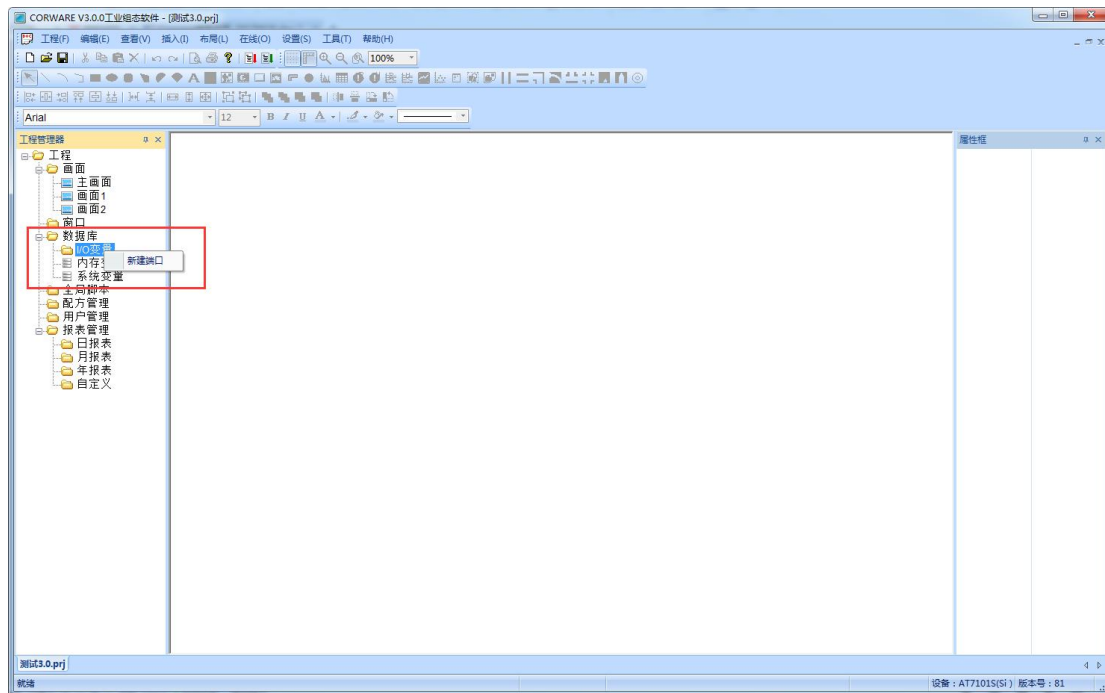
I/O 变量 是指可与外部数据采集程序直接进行数据交换的变量

“数据库”拥有“I/O 变量”，“I/O 变量”拥有“端口”，“端口”拥有“设备”，“设备”拥有“变量”

新建 I/O 变量的一般步骤： 新建端口→新建设备→新建变量

2.2.1 新建端口

右键点击【工程管理器】【I/O 变量】，左键点击新建端口，弹出【端口属性】窗口，设置完成后点击【确定】。



注：选择设备类型中的型号时，请点选右边的下拉菜单，找出匹配的型号，即使默认的型号就是所需的型号，也要点选下拉菜单，重新选择一次。

串口：是指数据一位一位地顺序传送，其特点是通信线路简单，只要一对传输线就可以实现双向通信（可以直接利用电话线作为传输线），从而大大降低了成本，特别适用于远距离通信，但传送速度较慢。

波特率：可以通俗的理解为一个设备在一秒钟内发送（或接收）了多少码元的数据。

奇偶校验：是一种校验代码传输正确性的方法。根据被传输的一组二进制代码的数位中“1”的个数是奇数或偶数来进行校验。采用奇数的称为奇校验，反之，称为偶校验。采用何种校验是事先规定好的。

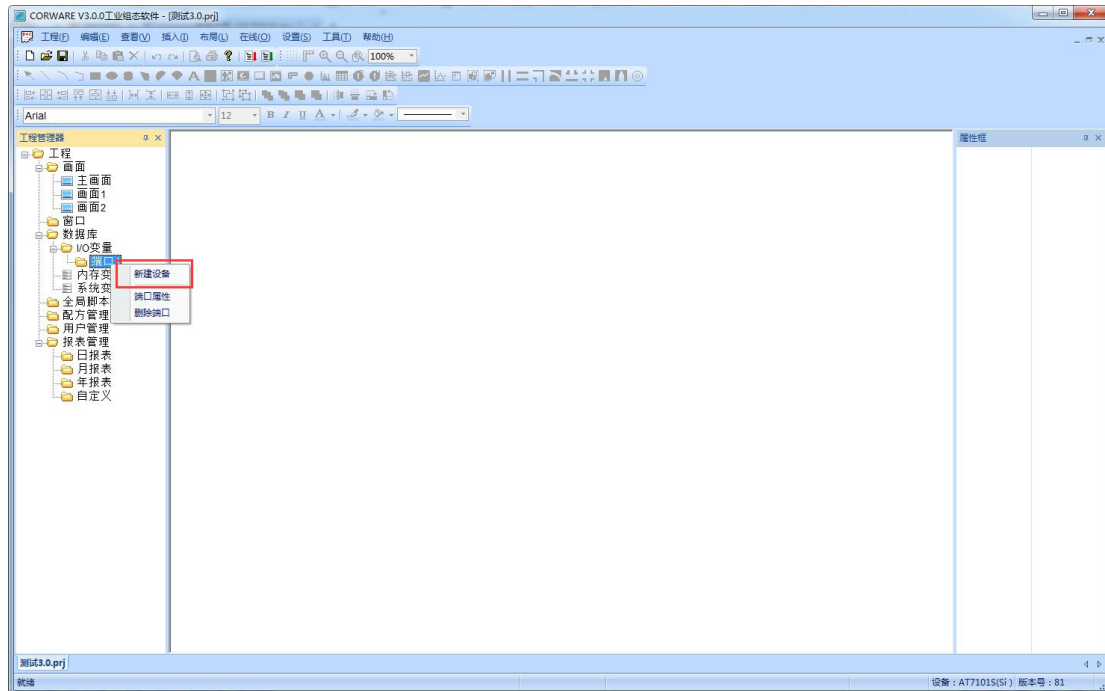
数据位：表示一组数据实际包含的数据位数，传输的数据格式由起始位、数据位、奇偶校验位和停止位组成。

端口名称	可自行定义
设备厂家和设备类型	PLC 等下位机厂家和型号，此选项将关联设备驱动 注意：设备类型若自动显示为所需型号，也须点下拉菜单选择。
串口参数	与 PLC 等下位机参数保持一致，其中串口号是指 HMI 中的串口编号。

Modbus 从站：打开菜单栏中的工具-----工程设置-----modbus 从站设置

2.2.2 新建设备

右键点击【端口】，在下拉框中左键点击【新建设备】，弹出【设备属性】窗口，设置完成后点击【确定】。

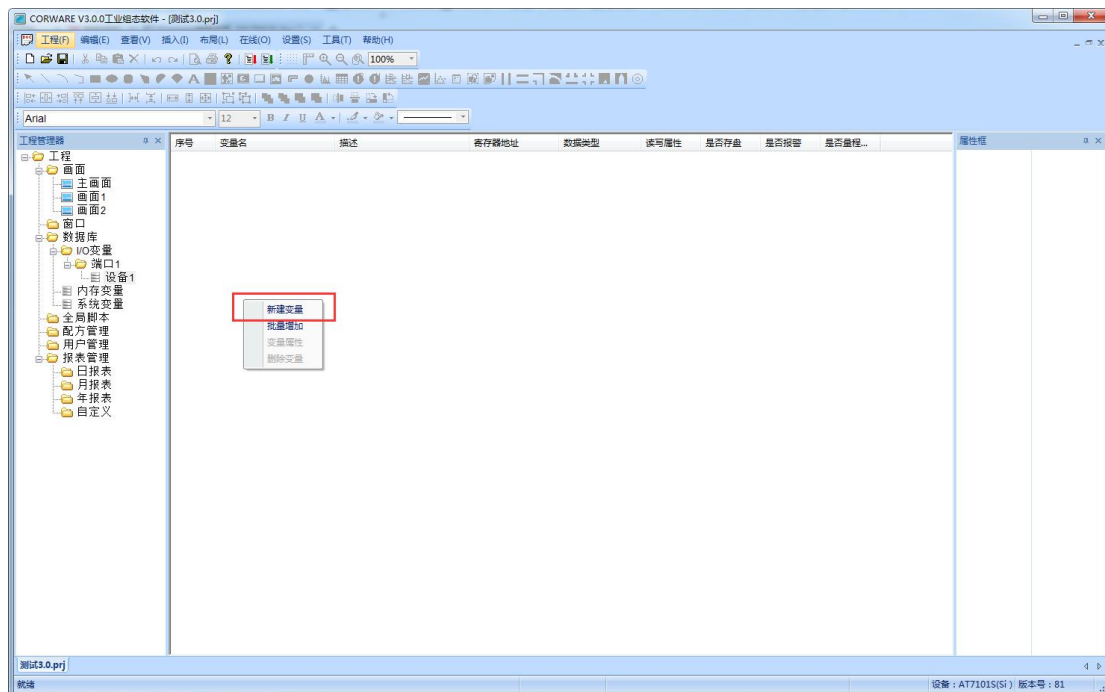




设备名称	PLC 等下位机设备，名称可自行定义
设备地址	设备的站号，需与下位机地址保持一致。

2.2.3 新建变量

左键单击“设备 1”，在右侧编辑框中右键单击，在弹出列表中选择【新建变量】，设置变量属性，点击确定。



变量属性

基本属性 存盘属性 报警属性 里程变换

名称:

描述:

寄存器类型: 数据类型:

数据块块号: 寄存器地址: 数据位:

只读 只写 原始值是码值 操作记录

确定 取消

名称	可自行定义，变量名需以字母开头。
寄存器类型	PLC 寄存器类型
数据类型	变量的数据类型
寄存器地址	变量在寄存器中的地址
数据位	变量在寄存器中地址的数据位

存盘属性

变量属性

基本属性 存盘属性 报警属性 里程变换

不存盘

定时存盘 存盘周期 秒

数据变化存盘 存盘精度 %

确定 取消

如果变量需要存盘，则点选定时存盘，存盘周期中输入数值。
报警属性

The screenshot shows the '报警属性' (Alarm Properties) tab of the '变量属性' (Variable Properties) dialog box. The '报警设置' (Alarm Settings) section contains the following options:

- 报警类型** (Alarm Type):
 - 不报警 (No Alarm)
 - 0->1报警动作, 1->0报警复归 (0->1 alarm action, 1->0 alarm reset)
 - 1->0报警动作, 0->1报警复归 (1->0 alarm action, 0->1 alarm reset)
 - 变位报警 (Position change alarm) with input fields for '0状态描述' (0 state description) and '1状态描述' (1 state description).
- 报警等级** (Alarm Level): A dropdown menu currently set to '一般' (General).

Buttons for '确定' (OK) and '取消' (Cancel) are located at the bottom right.

The screenshot shows the '报警属性' (Alarm Properties) tab of the '变量属性' (Variable Properties) dialog box, displaying a table of alarm settings:

报警类型	报警动作限值	报警等级	其他参数
<input type="checkbox"/> 下下限报警	0	一般	越限死区 0
<input type="checkbox"/> 下限报警	0	一般	
<input type="checkbox"/> 上限报警	0	一般	
<input type="checkbox"/> 上上限报警	0	一般	
<input type="checkbox"/> 大偏差报警	0	一般	偏差死区 0, 目标值 0
<input type="checkbox"/> 小偏差报警	0	一般	
<input type="checkbox"/> 变化率报警	0	一般	时间单位 秒

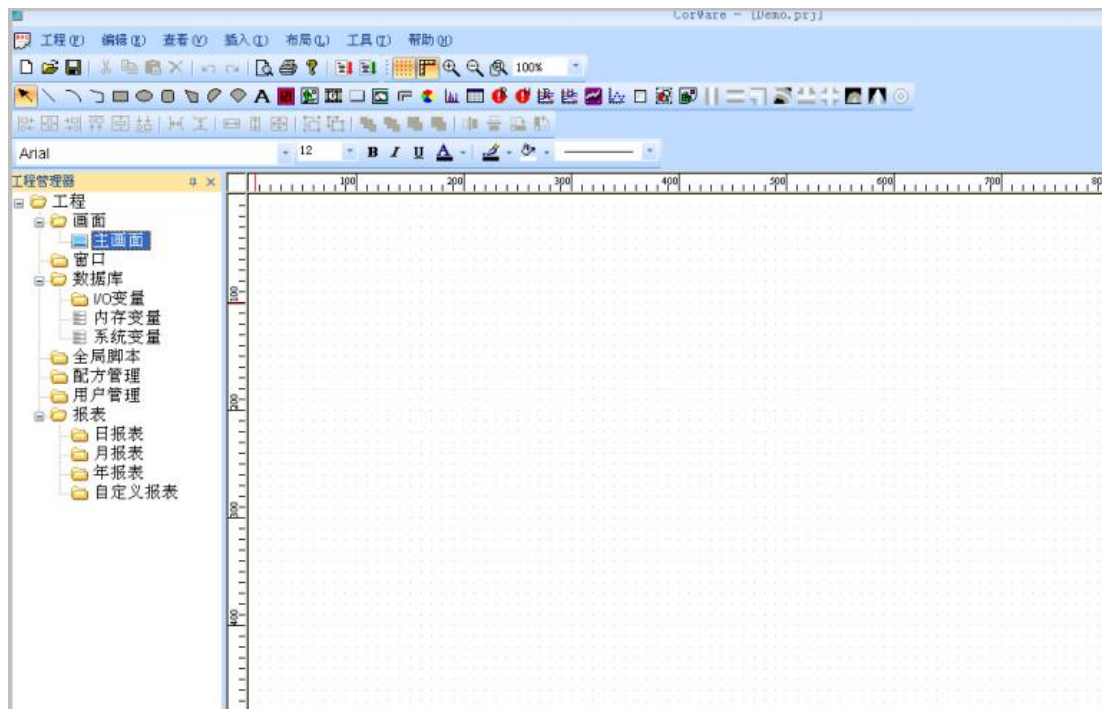
Buttons for '确定' (OK) and '取消' (Cancel) are located at the bottom right.

量程变换

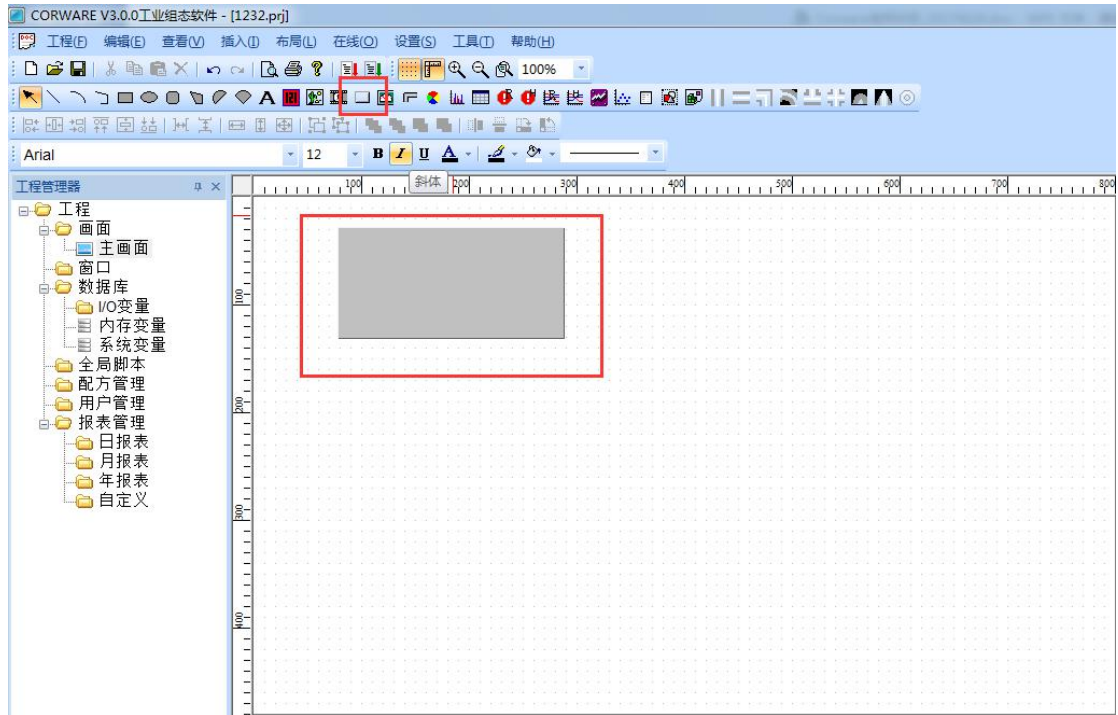


2.3 创建组态画面

2.3.1 点击左侧【工程管理器】【主画面】



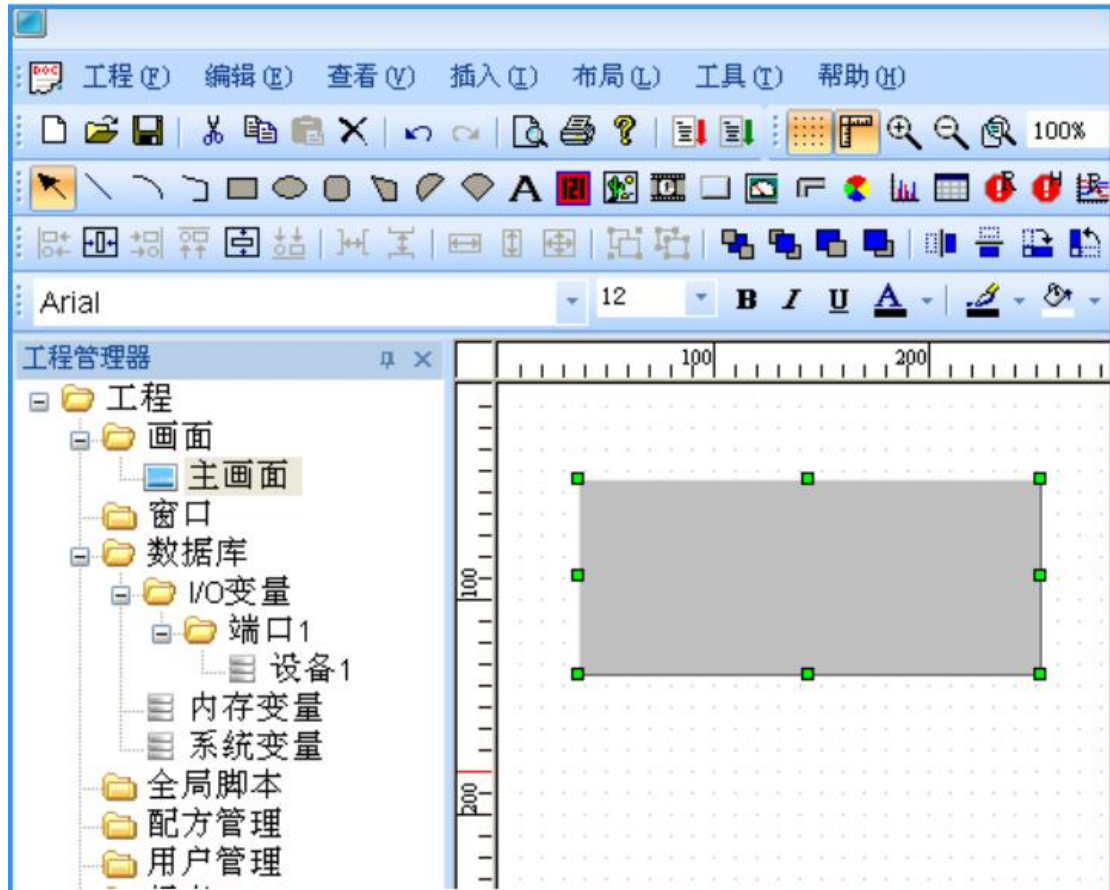
2.3.2 点击【按钮】图标，在编辑区域拖出一个矩形。



2.4 建立动画连接

2.4.1 点击编辑框中新建的按钮图标



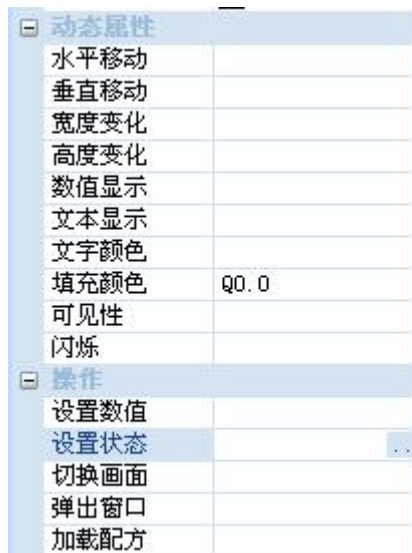


2.4.2 在右侧属性框中，单击【动态属性】【填充颜色】右侧小按钮，弹出‘颜色变化’窗口，设置颜色变量为 Q0.0，设置 Q0.0 不同值对应的不同颜色。





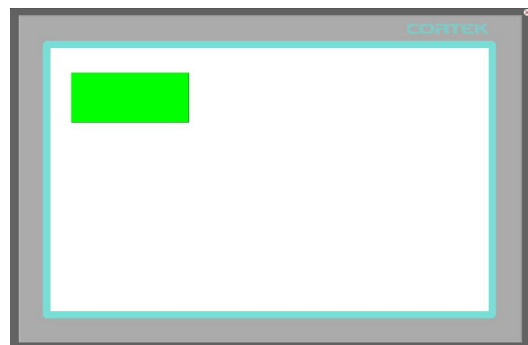
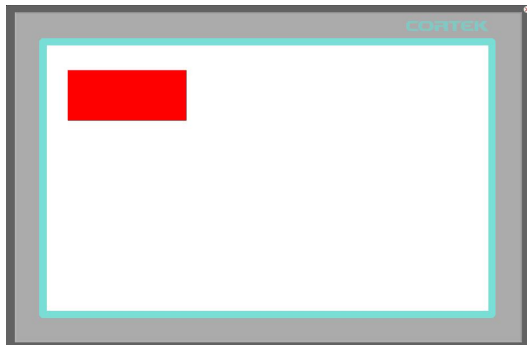
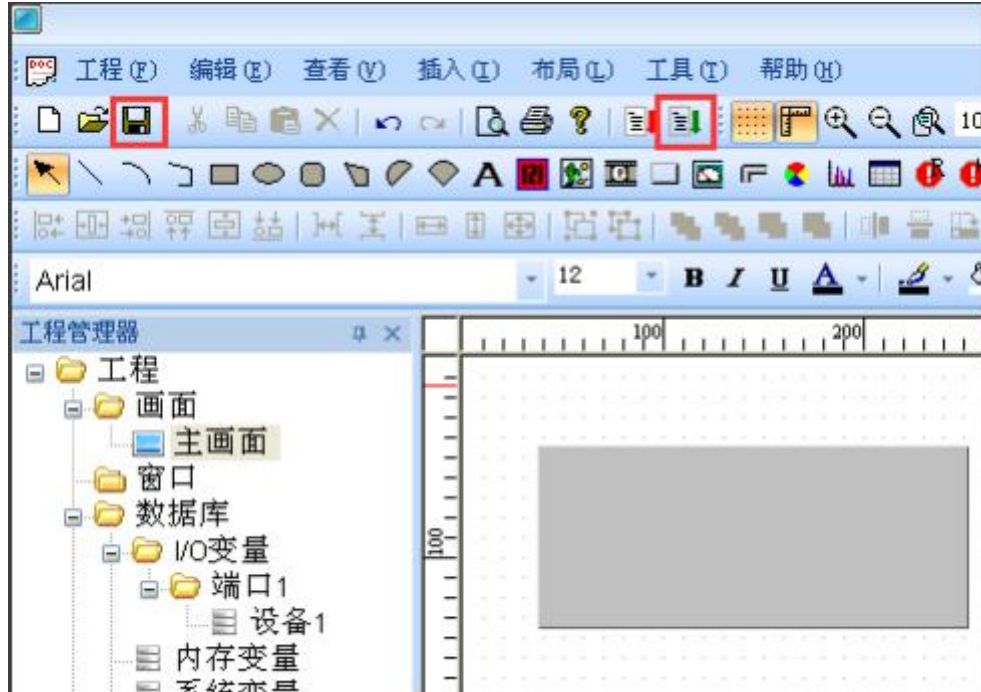


2.4.3 在右侧属性框中，单击【操作】【设置状态】右侧小按钮，弹出‘设置状态’窗口，点击右侧按钮选择状态变量为 Q0.0，操作类型选择取反。



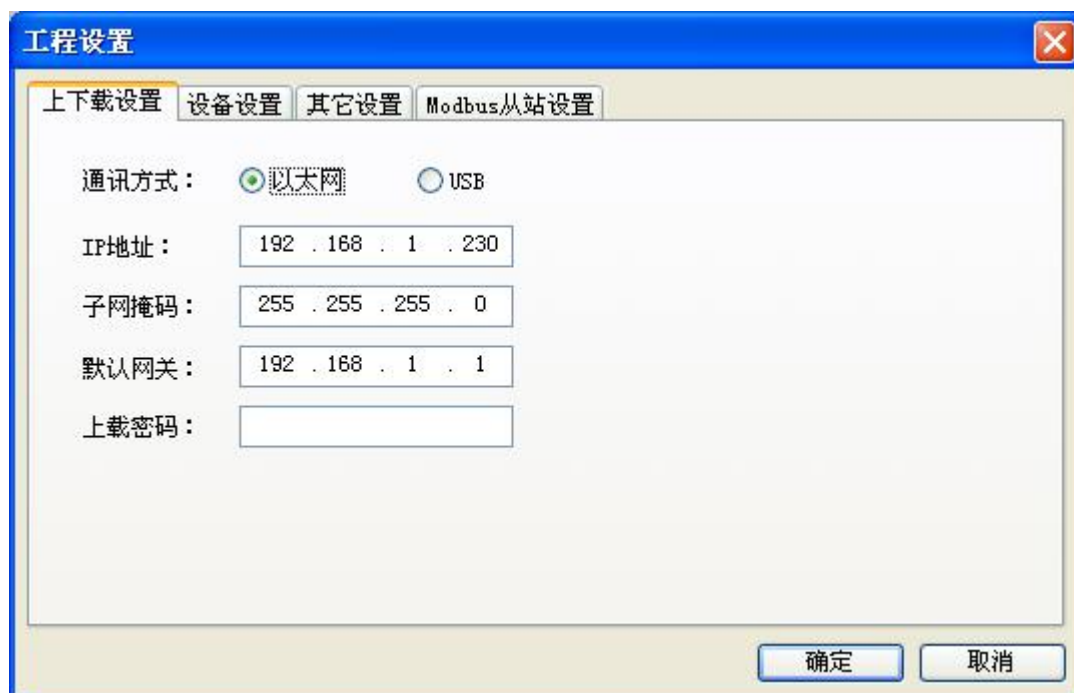
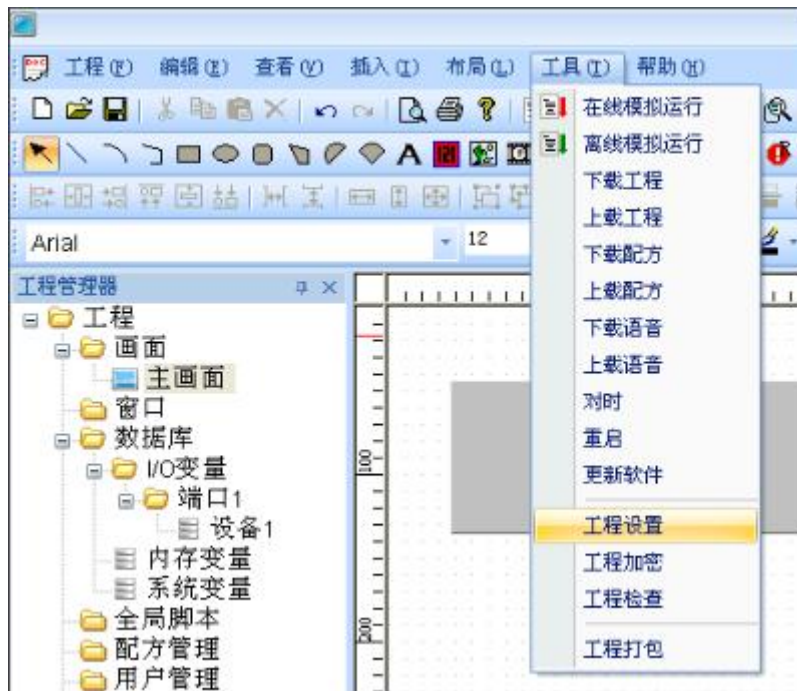
2.5 离线模拟

单击保存按钮，然后单击离线模拟按钮，PC 弹出模拟窗口，用鼠标点击窗口中按钮图标，按钮颜色会在红绿之间转换。

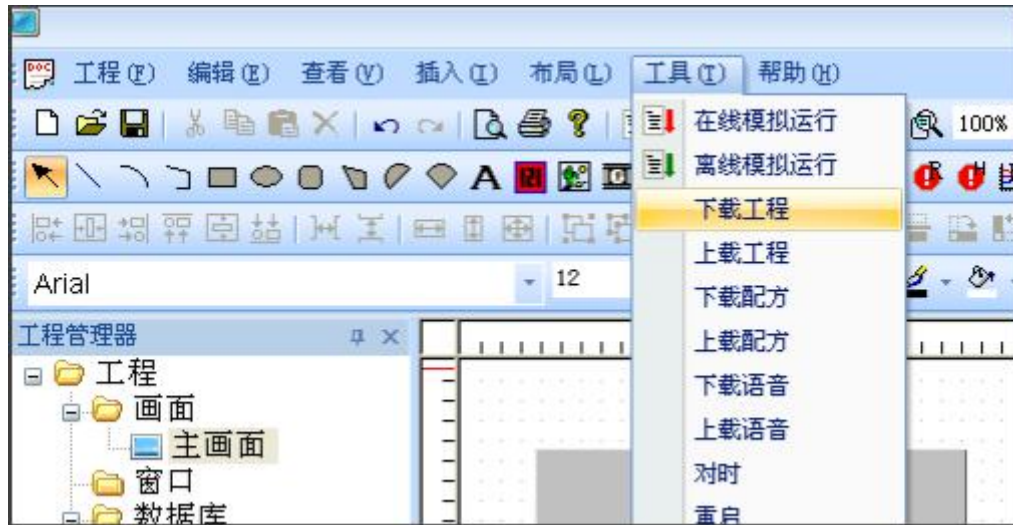


2.6 下载工程到 HMI

2.6.1 单击菜单栏中【工具】【工程设置】，在弹窗中设置通讯方式为以太网，设置对应的 IP（HMI 默认 IP：192.168.1.230），单击确定。本地 PC IP 地址须为和 HMI IP 同一网段的不同地址：例：
192.168.1.XX



2.6.2 HMI 上电后，用网线连接 PC 和 HMI，单击菜单栏中【工具】
【下载工程】，CORWARE 开始将驱动和工程文件下载到 HMI 中。



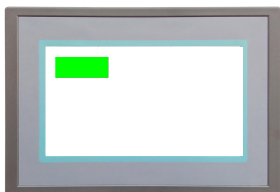
正在传输文件S7200PPIPLCDrv, 请稍候...



正在传输文件Demo.prj, 请稍候...




2.6.3 此时 HMI 中按钮图标为灰色，点击后显示“通讯故障，不能操作”，这是因为驱动未检测到对应的设备；用通讯线连接 PLC，此时即可通过 HMI 中按钮控制 PLC Q0.0 的通断，并可在 HMI 中通过颜色知道 Q0.0 此时的状态。



第三章 CORWARE 菜单栏详解

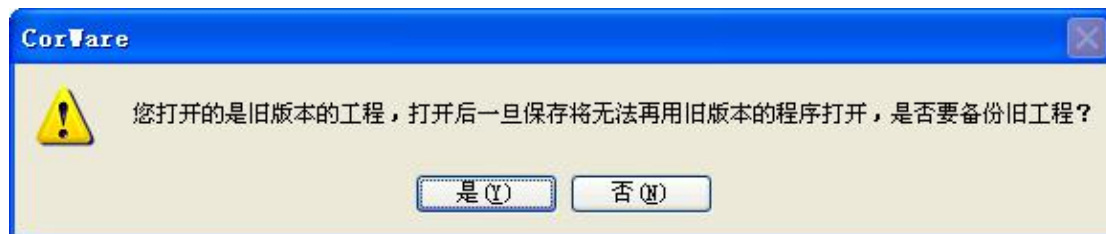
菜单栏包括：工程、编辑、查看、插入、布局、工具、帮助


3.1 工程菜单


 **新建工程**：根据工程向导，填入“工程名称”“工程路径”“设备型号”即可建立一个新的工程，快捷键 Ctrl+N

 **打开工程**：选择路径打开一个已经保存过的工程，快捷键 Ctrl+O

注意：软件各版本可以进行向上兼容。使用高版本可以升级打开低版本工程，低版本工程一旦升级打开之后，就不能使用低版本软件打开。因此用户在升级工程之前要做好工程备份。




 **保存工程**：保存对工程所做的修改，快捷键 Ctrl+S。（修改后若不保存工程，下载时将不包含修改的部分）

 **打印**：选择打印机、打印范围、份数等来打印当前的工程。打印时需先选中【工程管理器】【画面】，并且一次只能打印一个画面，快捷键 Ctrl+P。

3.2 编辑菜单

 **撤销**：取消以前执行过的命令，从最后一次操作开始，快捷键 Ctrl+Z。

 **重做**：重新执行先前已撤销的命令，快捷键 Ctrl+Y。

 **剪切**：删除当前画面中选中的一个或多个图形对象，并复制到粘贴缓冲区中，快捷键 Ctrl+X。

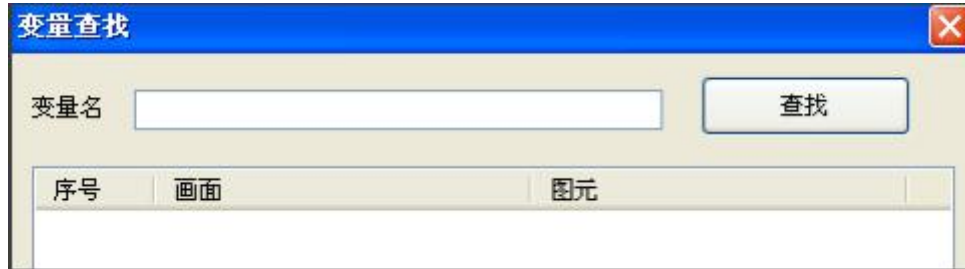
 **复制**：将当前选中的一个或多个图形对象复制到粘贴缓冲区中，快捷键 Ctrl+C。

 **粘贴**：将当前粘贴缓冲区中的一个或多个图形对象复制到指定位置，快捷键 Ctrl+V。

✕ 删除：删除一个或多个选中的图形对象，快捷键 Del。

全选：使画面上所有对象都处于选中状态，快捷键 Ctrl+A。
在画面中选择元件的同时，按下<Shift>键可以多选。

变量查找：通过变量名查找其关联的画面和图元。



变量替换：替换工程内变量名。



变量使用一览表：显示当前工程所使用的的所有变量，及对应的画面和图元。

序号	变量名	画面	图元
1	Q0.0	主画面	Obj2

文本替换：替换工程中所有相同的字符。



3.3 查看菜单

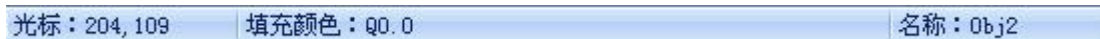
3.3.1 工具栏

显示或隐藏以下所示“常用工具栏”“绘图工具栏”“布局工具栏”“格式工具栏”。



3.3.2 状态栏

显示工具的用途、鼠标的坐标、元件关联的变量、名称等信息，状态栏在界面的最下方。



3.3.3 皮肤风格

选择不同风格的界面皮肤。



3.3.4 语言

中文和英文界面转换，切换语言之后需重新打开组态软件。



3.3.5 工程管理器

显示或隐藏工程管理器，工程管理器位于软件界面左侧。



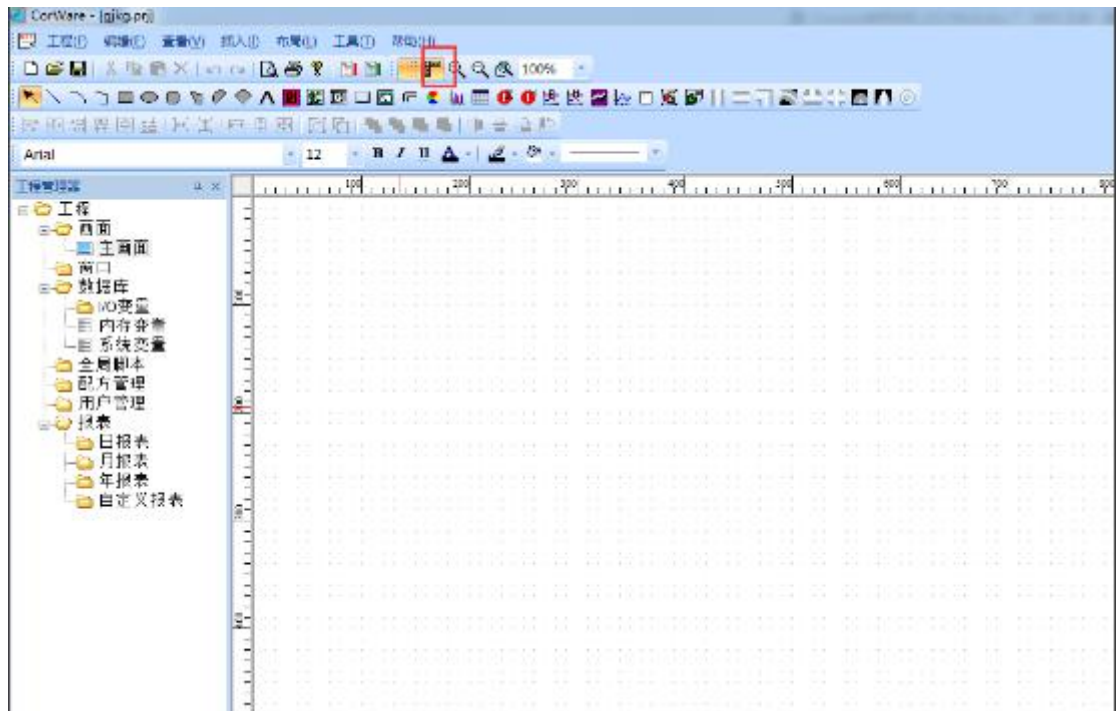
3.3.6 属性框

显示或隐藏属性框，属性框位于软件界面右侧。

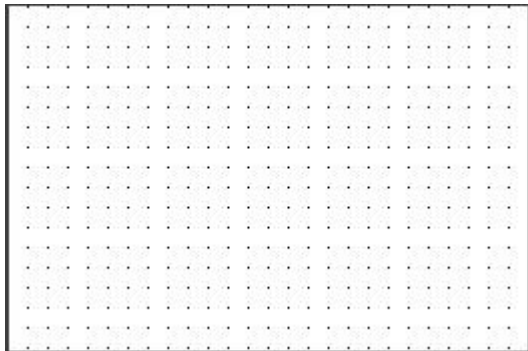


3.3.7 标尺

显示或隐藏标尺，标尺单位是像素。



3.3.8 网格：显示或隐藏网格。网格便于放置元件，在模拟运行或下载到 HMI 中后，网格不显示。



3.3.9 网格设置：设置网格颜色、间距、形状。

3.3.10 放大：放大显示编辑框及其中的元件。

3.3.11 缩小：缩小显示编辑框及其中的元件。

3.3.12 恢复：使编辑框恢复默认大小。

3.4 插入菜单

3.4.1 直线



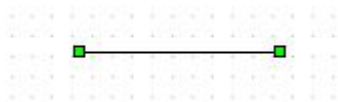
1. 点选“直线”工具后，在编辑框内点击左键不放拉出一条直线，放开左键完成。
画直线前按住“Shift”键不放可画出垂直或水平的直线

2. 当直线在选中状态时，右侧属性框中可以看到该图元的属性。

属性框	
基本属性	
名称	Obj8
X	162
Y	343
宽度	100
高度	0
可见性	只在本画面中...

名称	直线名称，可自行定义
X	直线左端顶点对应标尺的水平坐标
Y	直线左端顶点对应标尺的垂直坐标
宽度	直线水平尺寸
高度	直线没有高度，该值对矩形等有效
可见性	可选仅本画面中可见或所有画面可见

3. 在直线属性中可选择线条颜色和线条宽度



直线属性	
线条颜色	000000
线条宽度	1



直线属性	
线条颜色	00ff00
线条宽度	10

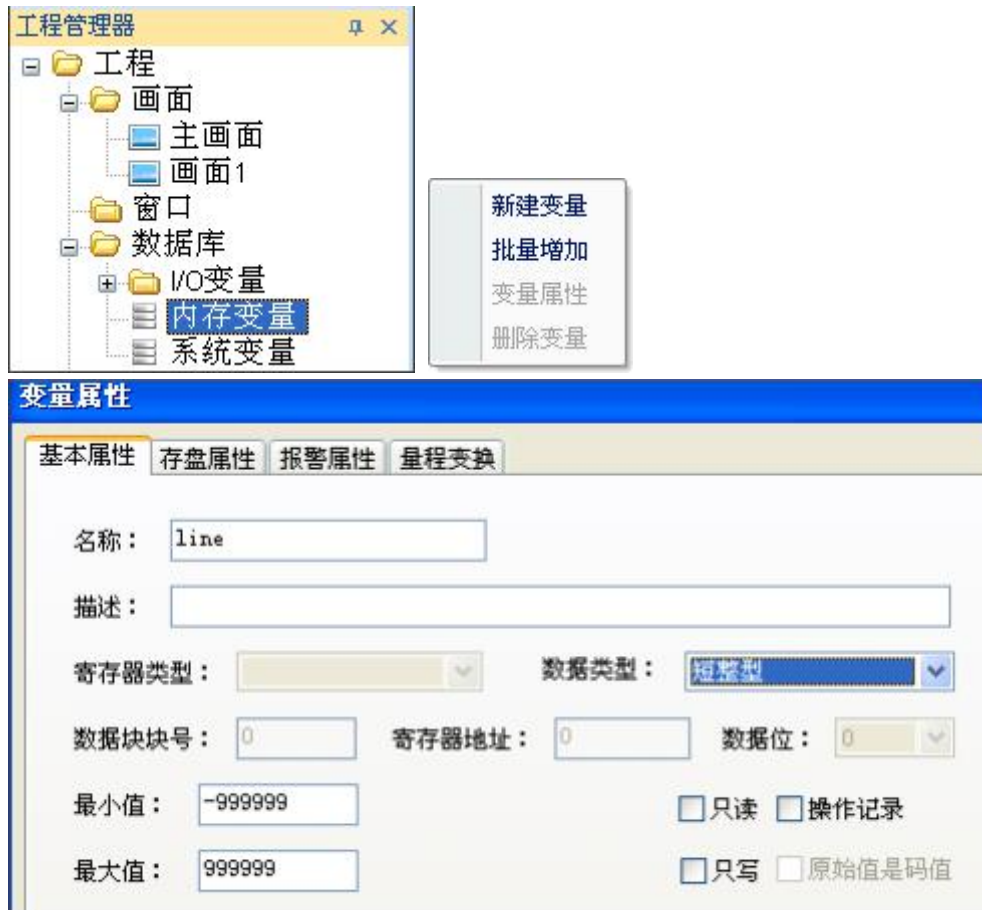
4. 动态属性（水平移动）

（元件的坐标、尺寸等关联变量，通过脚本实时改变变量数值，实现动态效果）

动态属性	
水平移动	
垂直移动	
宽度变化	
高度变化	
线条颜色	
可见性	
闪烁	

4.1 建立内存变量：

点选【工程管理器】【数据库】【内存变量】，在右侧编辑框内右键，点击新建变量，填入变量名称和变量类型，点击确定。



4.2 图元关联变量

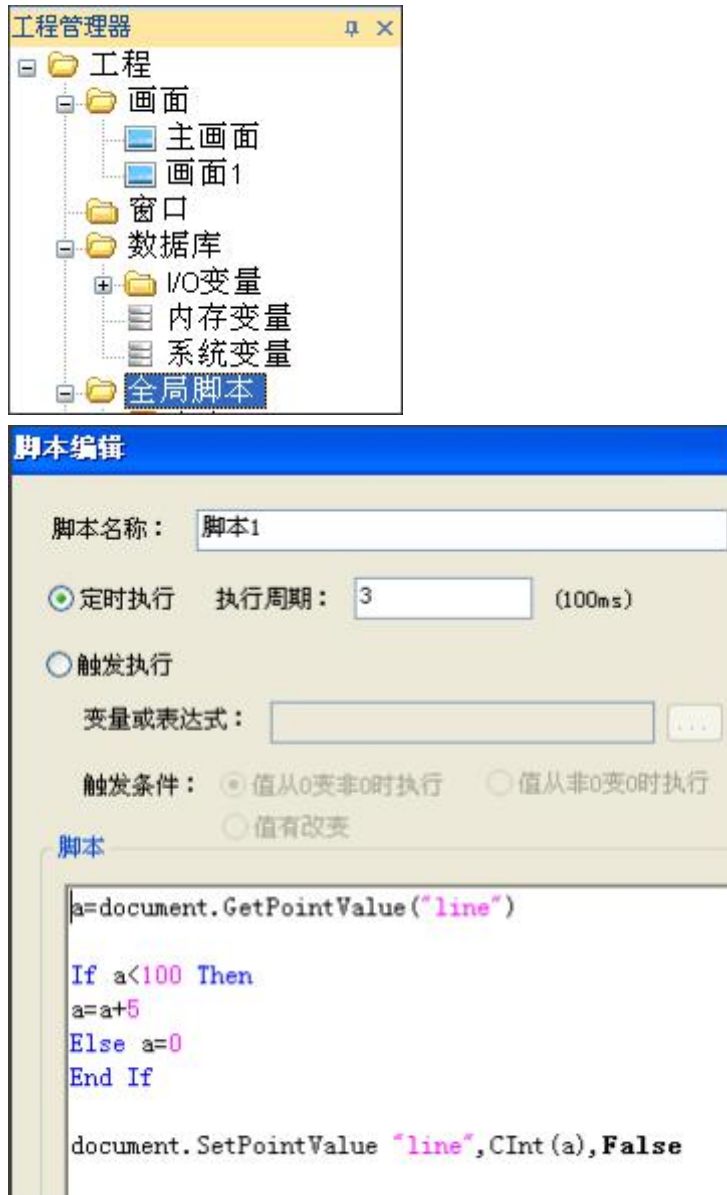
点击编辑框右侧【属性框】【动态属性】【水平移动】右侧的按钮，弹出“位置变化”窗口，点选变量“line”，填入对应的位置数值（位移值对应标尺数值，单位为像素）



最小位移	以元件左下角为原点的起始坐标	对应的值	变量最小值
最大位移	以元件左下角为原点的结束坐标	对应的值	变量最大值

4.3 建立脚本


右键【工程管理器】【全局脚本】，选择“新建脚本”，执行周期 300ms，输入代码。



`document.GetPointValue("line")`: 变量取值函数，可在右侧函数列表中调用。

`document.SetPointValue "line",CInt(a),False` : 变量赋值函数，可在右侧函数列表中调用。

注：输入函数的首字母，可在右侧函数列表中快速找到首字母开头的函数，大大的缩短了寻找函数的时间。

4.4 保存工程，点击菜单栏【工具】【离线模拟运行】，或直接单击工具栏中“”，直

线做循环水平移动。

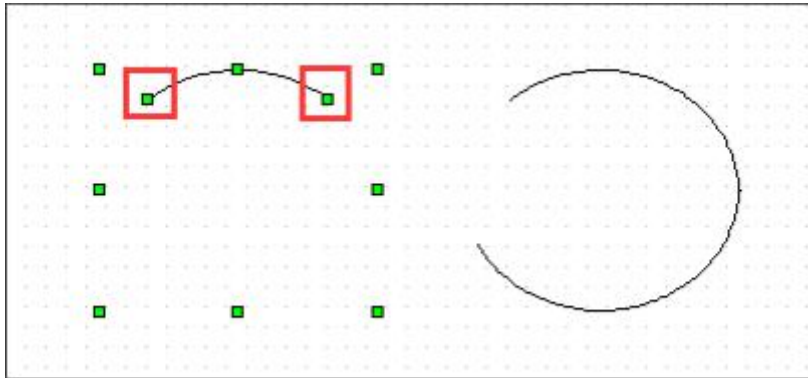
垂直移动、宽度变化、高度变化、线条颜色、可见性、闪烁 的用法与水平移动类似

3.4.2 弧



点选“弧”工具后，在编辑框内点击左键不放拉出一条弧线，放开左键完成。

拖动红框内绿色端点可调整弧的形状，得到类似右边的弧，拖动其它端点可调整弧的纵横比例。属性框设置可参照直线。



3.4.3 折线



点选“折线”工具后，在编辑框内点击左键放置一个端点，松开左键后将鼠标移到第二个端点处点击左键放置，类推放置其它端点，放置完成后右键完成，属性框设置可参照直线。

3.4.4 矩形



点选“矩形”工具后，在编辑框内拖出矩形，属性框设置可参照直线。

3.4.5 椭圆

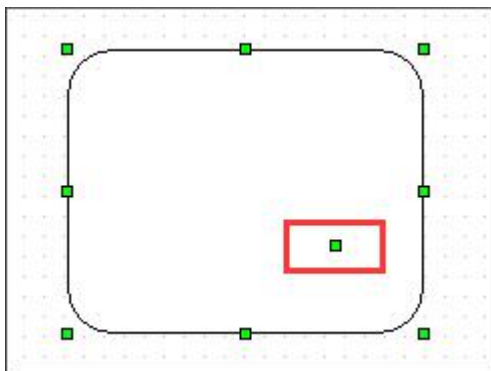


点选“椭圆”工具后，在编辑框内拖出矩形，椭圆内切该矩形，属性框设置可参照直线。

3.4.6 圆角矩形



点选“圆角矩形”工具后，在编辑框内拖出圆角矩形，图中红框标识的端点可调整圆角半径。属性框设置可参照直线。



3.4.7 多边形



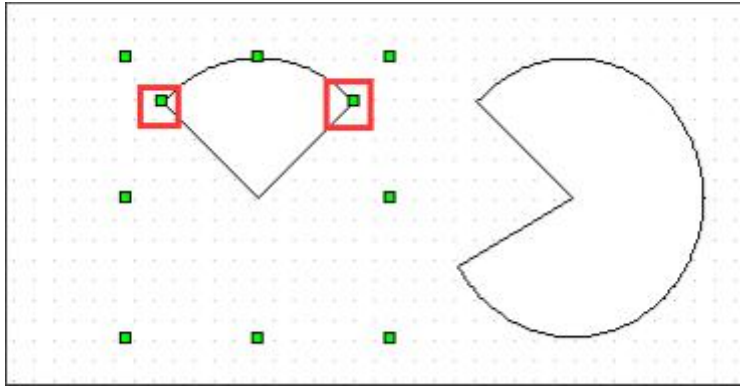
点选“多边形”工具后，在编辑框内左键放置多变形顶点即可，右键完成放置，属性框设置可参照直线。

3.4.8 弦

点选“弦”工具后，在编辑框内点击左键不放拉出一条弦，放开左键完成。拖动绿色端点可调整弦的形状，属性框设置可参照直线。

3.4.9 饼

点选“饼”工具后，在编辑框内点击左键不放拉出一个饼图，放开左键完成。拖动红框内绿色端点可调整饼图的形状，得到类似右边的饼图。属性框设置可参照直线。



3.4.10 文本

文本可显示常量或变量的值。

点选“文本”工具后，在编辑框内点拖出一个矩形框。

常量显示：【属性框】【文本属性】中设置文本内容和字符格式；

变量显示：【属性框】【动态属性】中【文本显示】和【数值显示】，关联变量后实时显示变量的值，或根据变量阈值不同显示不同的内容。

水平移动、垂直移动、宽度变化、高度变化设置请参照[直线](#)

文本属性	
文本	Text
字体	Arial (12)
文字颜色	000000
对齐	左对齐
动态属性	
水平移动	
垂直移动	
宽度变化	
高度变化	
数值显示	
文本显示	
文字颜色	
可见性	
闪烁	

数值显示	显示变量的数值，可添加前后缀
文本显示	显示变量的文本或根据变量阈值显示文本
文字颜色	根据变量阈值不同，文字颜色不同
可见性	变量数值为 0 或非 0 实现文字是否可见
闪烁	变量数值为 0 或非 0 实现文字是否闪烁

数值显示：显示变量的数值

整数位数	默认为0，设置为其它数时，实际数值位数不够时在整数之前补零，例：整数位	小数位数	输入范围是0--4位数，例如小数位数是2，变量值52，则显示数值是52.00
前缀字符	在变量显示值的前面加入前缀字符	后缀字符	在变量显示值的后面加入后缀字符

文本显示：根据变量阈值不同，显示不同文本



文字颜色:



可见性:



闪烁:



3.4.11 数据输入框

使用方法与文本类似，增加线条颜色和填充颜色，通过变量阈值不同变换不同颜色。

3.4.12 位图

点击位图工具，在弹出的窗口中选择本地 PC 路径中图片（支持 jpg 和 bmp 格式），确定后在编辑框中用鼠标左键拉出一个矩形，选择的图片内嵌在该矩形中，调整矩形大小可调整图片大小。属性框中【位图显示】，可通过变量阈值不同，显示不同的图片。



3.4.13 位图动画

点击位图工具，在弹出的窗口中选择本地 PC 路径中图片（支持 jpg 和 bmp 格式），确定后在编辑框中用鼠标左键拉出一个矩形，选择的图片内嵌在该矩形中，调整矩形大小可调整图片大小。点击【属性框】【位图动画属性】【设置】右侧按钮，在弹出的属性框中增加图片、设置播放速度，变量与播放条件组合后触发播放。





3.4.14 按钮

点击按钮工具，在编辑框中拖出一个矩形，即建立了一个按钮，动态属性可参照直线。

按钮属性	
填充颜色	 c0c0c0
线条宽度	1
文本	602j
字体	Arial (12)
文字颜色	 000000

填充颜色	按钮颜色选择
线条宽度	按钮边框外斜面宽度
文本	按钮上文字，可在此处直接输入
字体	按钮上文字字体选择
文字颜色	按钮上文字颜色选择

操作	
设置数值	
设置状态	
切换画面	
弹出窗口	
加载配方	

设置数值	替代按钮上文字，显示数值
设置状态	通过操作类型改变变量数值
切换画面	点击后切换到指定画面
弹出窗口	点击后弹出指定窗口
加载配方	加载指定的配方组

设置数值：选择变量后，选择操作类型

键盘输入	在点击按钮时弹出数值输入键盘，可更改按钮上显示的数值
设置常数	在点击按钮时数值变更为一固定常数
加	点击按钮一次自动加一固定常数
减	点击按钮一次自动减一固定常数
递加	按住按钮不松，持续加一固定常数，执行速度可设置
递减	按住按钮不松，持续减一固定常数，执行速度可设置
设置字符串	在点击按钮时弹出输入键盘，可更改按钮上显示的字符（设置字符串时，同时设置动态属性中【文本显示】变量）

注：如果变量输出不需要数值显示，则去掉 变量输出到显示 中的√。

设置状态：点击按钮后通过操作类型改变变量数值

设置状态

变量

操作类型

操作权限

取消

切换画面：点击按钮后切换到指定画面

切换画面

画面名称

操作权限

确定

取消

弹出窗口

窗口名称

窗口位置

X 0 Y 0

窗口标题栏

显示标题栏

操作权限

确定

取消

加载配方：加载配方组中的配方，详见【工程管理器】【配方管理】



3.4.15 仪表

点击仪表工具，在编辑框中拖出一个仪表图案。

变量或表达式	刻度所关联的变量或表达式
量程范围	变量数值范围对应的角度范围
刻度	仪表刻度线颜色及形状设置
指针	仪表指针颜色及形状设置
越限报警线	在仪表外围用指定颜色标识报警范围
刻度标签	刻度数值格式设置
标尺	仪表外框



3.4.16 管道

点击仪表工具，按住“shift”键不放，在编辑框中点击左键确认第一点，管道延伸到垂直或水平第二点时点击左键确认，依次完成管道绘制，点击右键退出。在属性框中设置“管道内壁颜色”“管道宽度”“管道流动设置”。

“管道中央颜色”在流动管道中无效，管道流动设置中变量触发流动块是否流动。

管道属性	
管道中央颜色	<input type="checkbox"/> fffffff
管道内壁颜色	<input checked="" type="checkbox"/> 808080
管道宽度	12
流动设置	...

管道流动设置
✕

流动块

颜色

样式 椭圆

块间间隔

长度

流动方向

正向 反向

流动速度

快速

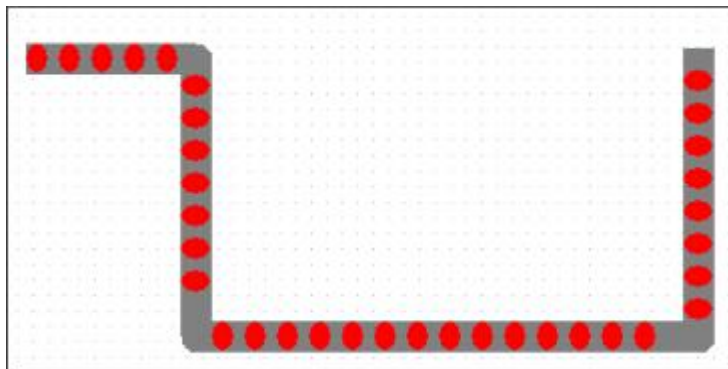
中速

低速

静态是否可见

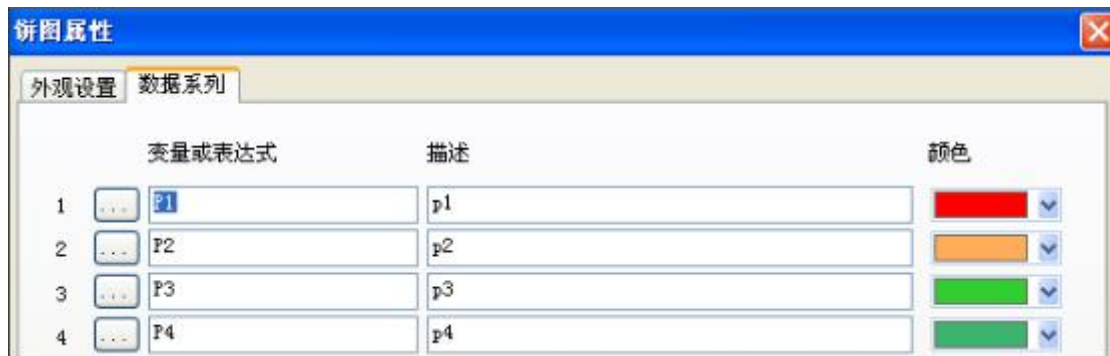
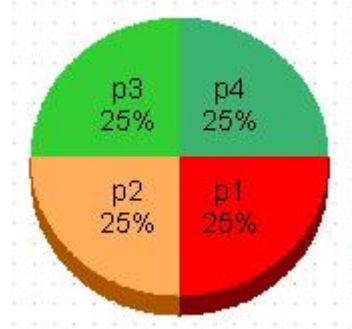
可见 不可见

流动控制变量或表达式



3.4.17 饼图

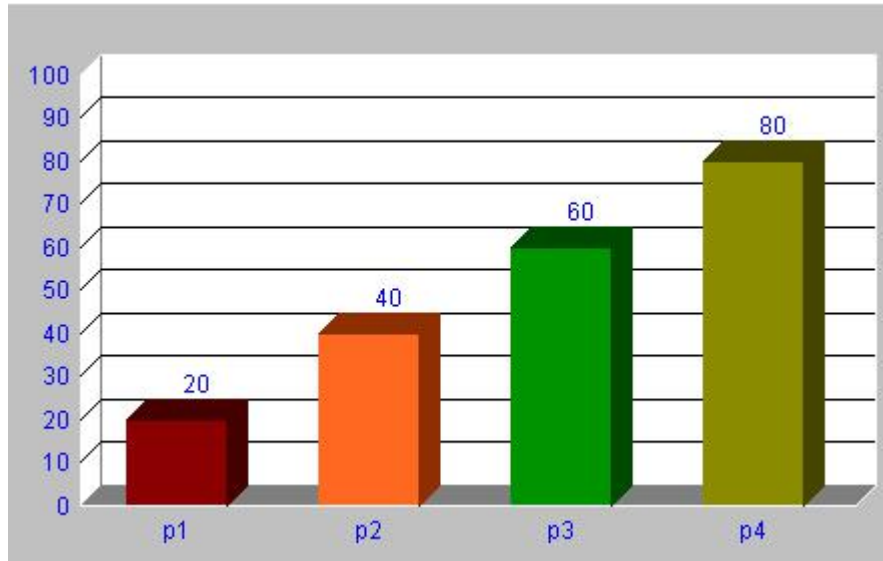
点击饼图工具，在编辑框内拖出一个饼图，点击【属性框】【饼图属性】【设置】



数据标签	饼图上字符格式设置
立体度	饼图倾斜度
数据系列	数据变量描述及对应颜色

3.4.18 棒图

点击棒图工具，在编辑框内拖出一个棒图，点击【属性框】【棒图属性】【设置】



棒图属性

外观设置 | 数据系列

背景色 最大值
 小数位 最小值
 3D显示方式 立体度

棒图区
 背景色 边框色

网格
 颜色 数目

显示当前值
 颜色 字体

名称标签
 颜色 字体

刻度标签
 颜色 字体

棒
 边框色 无 宽度
 填充方式 单色 渐变色

确定 取消

棒图属性

外观设置 | 数据系列

	变量或表达式	描述	颜色
1	<input type="text" value="P1"/>	p1	<input type="text" value=""/>
2	<input type="text" value="P2"/>	p2	<input type="text" value=""/>
3	<input type="text" value="P3"/>	p3	<input type="text" value=""/>
4	<input type="text" value="P4"/>	p4	<input type="text" value=""/>

3.4.19 报表

点击报表工具，在编辑框内拖出一个报表，通过函数“ReportLoad”加载【工程管理器】【报表】中建立的报表，通过函数“ReportQuery”查询报表历史数据，详见 page 报表系统。

3.4.20 实时报警

点击实时报警工具，在编辑框中拖出一个报警显示列表，当变量超出阈值时，实时报警列表中会实时显示该报警的“时间”“描述”“类型”“当前值”“限值”等。

3.4.21 历史报警

点击历史报警工具，在编辑框中拖出一个报警显示列表，点击表中【查询条件】，选择时间段，即可在列表中显示该段时间内所有报警的信息。

3.4.22 实时曲线

点击实时曲线工具，在编辑框中拖出一个曲线控件，在属性框中设置曲线格式及关联变量后，即可显示以时间为 X 轴，数据值为 Y 轴的实时曲线。

3.4.23 历史曲线

点击历史曲线工具，在编辑框中拖出一个曲线控件，在属性框中设置曲线格式及关联变量，点击控件中【查询条件】选择时间段，即可显示以时间为 X 轴，数据值为 Y 轴的历史曲线。

3.4.24 计划曲线

点击计划曲线工具，在编辑框中拖出一个曲线控件，在属性框中设置曲线格式及关联变量，点击控件中【启动】按钮，即可查看计划与实际是否一致。

3.4.25 XY 曲线

点击计划曲线工具，在编辑框中拖出一个曲线控件，在属性框中设置曲线格式，通过脚本设置 x、y 的值描点曲线。

3.4.26 变量浏览

点击变量浏览工具，在编辑框中拖出一个变量浏览控件，在属性框中设置表格格式，运行时可显示所有变量的‘变量名’‘描述’‘当前值’

3.4.27 图符

点击图符工具，在弹窗中选中图符组下的图符，点击确定，在编辑框中拖出一个图符。选中新建的图符，在【属性框】【动态属性】【图符显示】窗口中关联变量，以显示图符不同的状态（图符初始状态对应的变量值为 0）。



3.4.28 插件

点击插件工具，在弹出的工具列表中包含“数字时钟”和“数码管”；

“数字时钟”可实时显示当前的系统时间，“数码管”可实时显示关联变量数值；选中“数字时钟”，点击确定，在编辑框中拖出一个“数字时钟”，右侧【属性框】【数字时钟属性】中可设置格式和颜色。“数码管”插件的使用同“数字时钟”。



3.4.29 管道

管道工具包含各种形状，选择对应的管道类型，在编辑框中拖出一个管道，【属性框】【管道属性】中可更改管道颜色。通过工具栏中“布局”“组合”工具，可将多种管道组合为一个元件，从而组成各种不同的形状。

3.5 布局菜单



3.5.1 叠放次序

包含【移至底层】【移至顶层】【下移一层】【上移一层】；选中单一元件后，点击叠放次序中的工具，即可改变相交元件的叠放次序。

3.5.2 对齐

包含【左对齐】【水平居中】【右对齐】【顶端对齐】【垂直居中】【底端对齐】；选中 ≥ 2 个图元，点击对齐工具即可对齐图元；其中【水平居中】和【垂直居中】可作用于单一图元。

3.5.3 均匀分布

包含【垂直均匀分布】【水平均匀分布】；选中 ≥ 2 个图元，点击均匀分布工具即可对齐图元；

3.5.4 统一尺寸

包含【相同宽度】【相同高度】【相同宽度和高度】；
选中 ≥ 2 个图元，点击统一尺寸工具即可改变图元尺寸；

3.5.5 组合

包含【组合】和【取消组合】；

选中 ≥ 2 个图元，点击【组合】工具，可将选中的多个图元组合成单一图元；
选中组合后的图元，点击【取消组合】工具，图元被还原成组合前的多个图元。

3.5.6 翻转

包含【水平翻转】和【垂直翻转】；

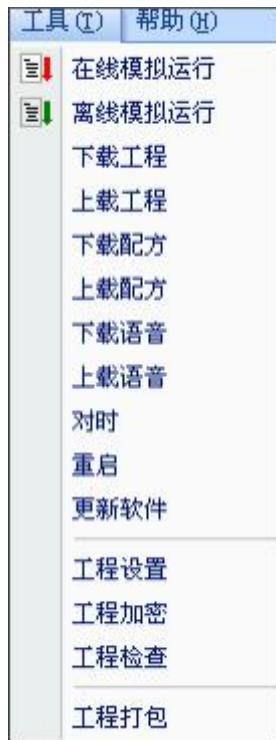
选中单一元件后，点击翻转工具，元件以镜像显示。

3.5.7 旋转

包含【顺时针旋转 90 度】【逆时针旋转 90 度】；

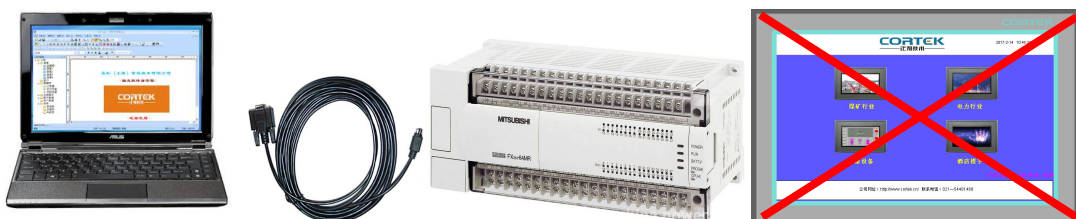
选中单一元件后，点击旋转工具，元件顺时针或逆时针旋转 90 度。

3.6 工具菜单



3.6.1 在线模拟运行

在 PC 上模拟工程的运行（PC 直接连接 PLC）；
不需要连接到 HMI，直接可在 PC 模拟实际运行。



3.6.2 离线模拟运行

在 PC 上模拟工程的运行

不需要连 PLC 和 HMI，大幅缩短编辑下载测试时间。

模拟运行时会自动按 HMI 型号模拟出 HMI 外形。



3.6.3 下载工程

下载工程到 HMI，1.在【工程设置】中设置 HMI IP 地址

2.网线连接 PC 与 HMI

3.点击【下载工程】即可

注意：本地 PC IP 地址须与 HMI IP 同一网段，例：192.168.1.XX



3.6.4 上载工程

上载 HMI 中工程到本地 PC，备份 HMI 中的工程。

1.在【工程设置】中设置 HMI IP 地址

2.网线连接 PC 与 HMI

3.点击【上载工程】，在上载工程窗口中设置“工程名称”“工程路径”“IP 地址”



3.6.5 下载配方

下载配方到 HMI 中

- 1.在【工程设置】中设置 HMI IP 地址
- 2.网线连接 PC 与 HMI
- 3.点击【下载配方】，即可将配方下载到 HMI 中，配方使用请参照【工程管理器】【配方管理】
- 4.重启 HMI 配方可生效。

3.6.6 上载配方

上载 HMI 中配方到 PC，备份 HMI 中的配方。

- 1.在【工程设置】中设置 HMI IP 地址
- 2.网线连接 PC 与 HMI
- 3.点击【上载配方】，在上载工程窗口中设置“工程名称”“工程路径”“IP 地址”

3.6.7 下载语音

下载语音文件到 HMI 中，语音文件目录在本地 PC 工程路径“Audio”文件夹中。

3.6.8 上载语音

上传 HMI 中语音文件到 PC，备份 HMI 中的语音文件

3.6.9 对时

将 PC 系统时间同步到 HMI 中

- 1.在【工程设置】中设置 HMI IP 地址
- 2.网线连接 PC 与 HMI
- 3.点击【对时】即可

3.6.10 重启

重启 HMI

- 1.在【工程设置】中设置 HMI IP 地址
- 2.网线连接 PC 与 HMI
- 3.点击【重启】即可

3.6.11 更新软件

更新 HMI 中组态环境（该功能暂不能用，待完善）

3.6.12 工程设置

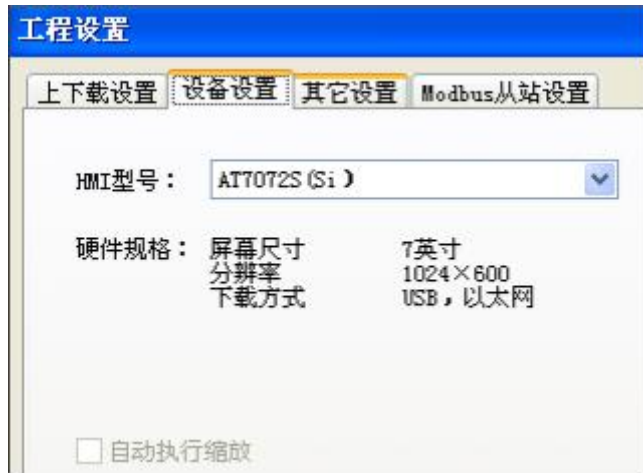
工程设置包含“上下载设置”“设备设置”“其它设置”“Modbus 从站设置”

上下载设置：上下载工程通讯方式设置

The screenshot shows a software configuration window titled "工程设置" (Engineering Settings). It has four tabs: "上下载设置" (Upload/Download Settings), "设备设置" (Device Settings), "其它设置" (Other Settings), and "Modbus从站设置" (Modbus Slave Settings). The "上下载设置" tab is active. It contains the following fields:

- 通讯方式 (Communication Method): Two radio buttons, "以太网" (Ethernet) is selected, and "USB" is unselected.
- IP地址 (IP Address): A text box containing "192 . 168 . 1 . 230".
- 子网掩码 (Subnet Mask): A text box containing "255 . 255 . 255 . 0".
- 默认网关 (Default Gateway): A text box containing "192 . 168 . 1 . 1".
- 上传密码 (Upload Password): An empty text box.

设备设置：设置 HMI 型号，若 HMI 分辨率与预设不同，可点选“自动执行缩放”。



其它设置:



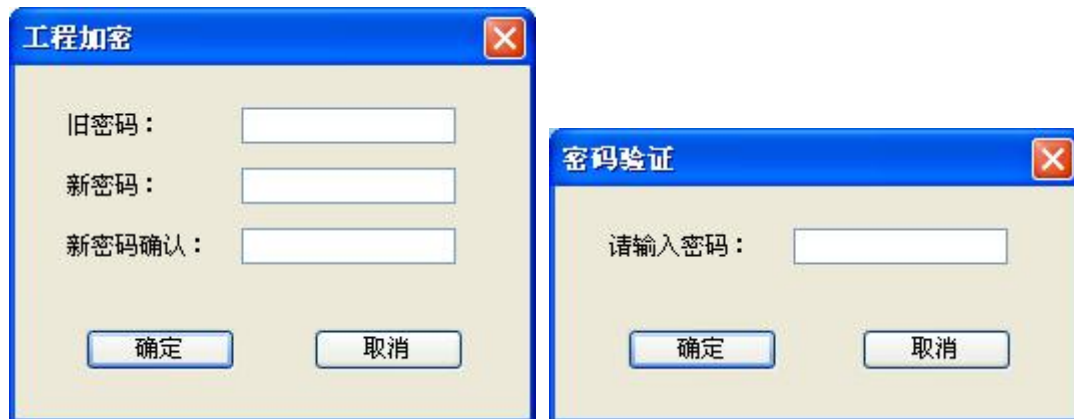
请注意当使用历史等数据时，请确定数据是否存储。一般都选择 FLASH 存储。菜单栏中的工具----工程设置----其它设置。如下图：



背光关闭等待时间	在设置时间内 若 HMI 处于空闲状态将关闭背光
画面刷新周期	HMI 画面更新周期设置
历史存储介质	历史数据存储方式设置
蜂鸣器	使能蜂鸣器
语音	使能语音

3.6.13 工程加密

设置工程密码，设置完成后，重新打开工程时需要输入密码，旧密码默认为空。



3.6.14 工程检查

检查工程是否存在错误。

3.6.15 工程打包

将工程文件存储为 hmi.prpj 文件，存放于 U 盘中，详见工程下载。

3.7 帮助菜单

“用户手册”，用户手册。

“VBScript 手册”，VBScript 语言中文帮助。

“关于”，CORWARE 相关信息。



3.8 菜单快捷方式

Alt+F 工程菜单

Alt+E	编辑菜单
Alt+V	查看菜单
Alt+I	插入菜单
Alt+L	布局菜单
Alt+T	工具菜单
Alt+H	帮助菜单
Ctrl+N	新建工程
Ctrl+O	打开工程
Ctrl+S	保存工程
Ctrl+P	打印
Ctrl+Z	撤消
Ctrl+Y	重做
Ctrl+X	剪贴
Ctrl+C	复制
Ctrl+V	粘贴
Del	删除
Ctrl+A	全选
Ctrl+F5	模拟运行

第四章 工程管理器

4.1 工程管理器概述

工程管理器的结构如图所示，类似于资源管理器的树状结构图，主要展示工程的各个组成部分。主要包括“画面”，“窗口”，“数据库”，“全局脚本”，“配方管理”，“用户管理”，“报表”；点击工程管理器右上角图标可以隐藏工程管理器，点击图标可以恢复。



4.2 画面

画面是 CORWARE 实时显示操作部分。在 CORWARE 运行时，工业现场的生产状况要以动画的形式反映在屏幕上，同时工程人员在计算机前发布的指令也要迅速送达生产现场，画面是联系操作人员和现场设备的界面，是操作人员的输入输出设备。

CORWARE 中画面从单个图元入手，可以在每个图元下关联相关的变量，进行变量的动态连接。

CORWARE 中工程管理器自动生成主画面。

4.2.1 新建画面：右键点击【画面】，选择【新建画面】，输入画面名称、是否为启动画面、背景颜色。

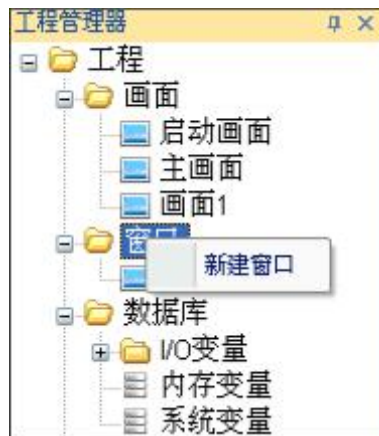


启动画面	启动画面用来做启动欢迎画面，可以设置画面显示时间，如果时间为0，则启动画面不显示。
主画面	主画面一般用于用户建立画面索引，在画面数量很多时，可以在主画面中将画面根据一定的原则进行管理。
正常画面	画面属性中不选择作为启动画面的画面为正常画面。

4.3 窗口

CORWARE 提供了可以设置宽度和高度的小画面，窗口用于用户自定义大小的画面调用。

新建窗口：右键点击【窗口】，选择【新建窗口】，输入窗口名称、标题、大小和背景颜色。



窗口调用：通过“按钮”等元件【操作】【弹出窗口】属性弹出窗口

也可以用函数 `openwindow`

操作	
设置数值	
设置状态	
切换画面	
弹出窗口	窗口1
加载配方	



弹出窗口配置对话框，包含以下配置项：

- 窗口名称**：下拉菜单，当前选择“窗口1”。
- 窗口位置**：X 坐标输入框为 55，Y 坐标输入框为 55。
- 窗口标题栏**：复选框“显示标题栏”已勾选。
- 操作权限**：下拉菜单，当前选择“0”。
- 底部有“确定”和“取消”按钮。

4.4 数据库

参照第五章【构造数据库】

4.5 全局脚本

全局脚本分为定时执行、触发执行两种动作。





脚本名称	可自行定义，标识代码功能
定时执行	间隔指定的时间执行一次脚本中的代码，单位为 0.1 秒
触发执行	通过变量和触发条件组合触发执行代码
脚本	待执行的代码

4.6 配方管理

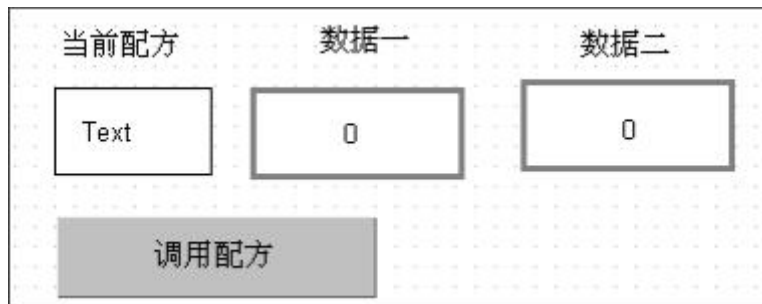
4.6.1 新建配方组：右键【配方管理】，左键单击【新建配方组】，在【配方编辑】

窗口中增加变量和配方，【确定】完成编辑



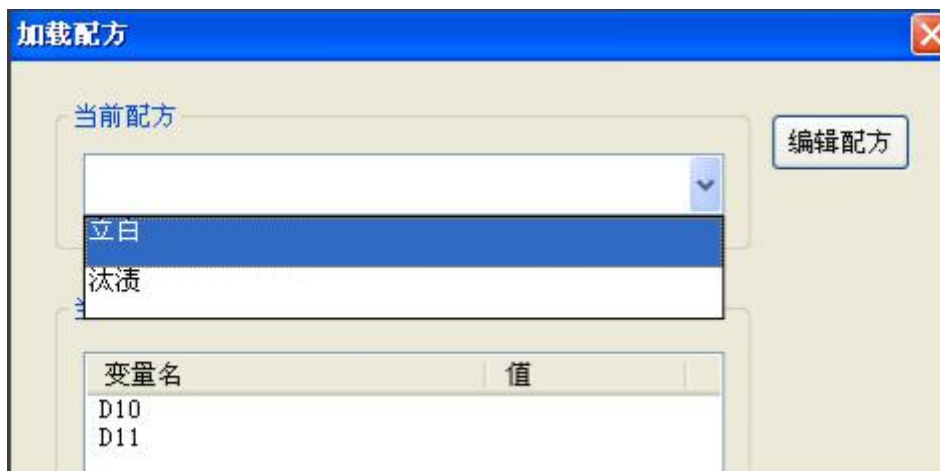


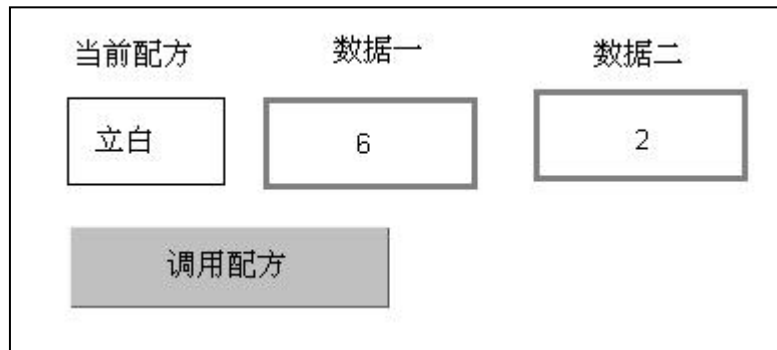
4.6.2 配方显示： 在编辑框中新建如下文本显示框和按钮。



当前配方	关联系统变量“\$CurrentRecipeName”，显示当前配方名
数据一	显示变量 D10 数值
数据二	显示变量 D11 数值
调用配方	属性框中点选【操作】【加载配方】加载配方组

4.6.3 配方加载： 运行时，点击【调用配方】按钮，弹出【加载配方】窗口，选择配方名称，【确定】后配方调用完成。





4.7 用户管理

4.7.1 点击【用户管理】，在编辑框中右键选择【新建用户】，在用户属性窗口中输入“用户名”“密码”“操作权限”；



4.7.2 系统将优先级从低级到高级定为 1 到 16，可以对用户、图形对象等设置不同的优先级。可定义操作优先级的为五种动作（设置数字、设置状态、切换画面、弹出窗口、加载配方）。

高级的权限可以打开低级的权限。



The image shows a Windows-style dialog box titled "用户属性" (User Properties). It has a blue title bar with a close button (X) in the top right corner. The main area is light beige and contains four input fields on the left and two buttons on the right. The input fields are labeled "用户名" (Username), "密码" (Password), "密码确认" (Confirm Password), and "操作权限" (Operation Permissions). The "操作权限" field is a dropdown menu. The buttons are labeled "确定" (OK) and "取消" (Cancel).

用户名	<input type="text"/>	确定
密码	<input type="password"/>	取消
密码确认	<input type="password"/>	
操作权限	<input type="text"/>	

4.8 报表。

详见 Page95 ”报表系统”

第五章 构造数据库

数据库是“CORWARE”最核心的部分。在 CORWARE 运行时，工业现场的生产状况要以动画的形式反映在屏幕上，同时工程人员在计算机前发布的指令也要迅速送达生产现场，所有这一切都是以实时数据库为中介环节，数据库是联系上位机和下位机的桥梁。

CORWARE 中数据库从设备入手，在每个设备下定义相关的变量，与传统组态软件建立设备后，变量在一张数据表中相比，更加直观，查找变量更加方便。

在 CORWARE 中，“数据库”拥有“I/O 变量”，“I/O 变量”拥有“端口”，“端口”拥有“设备”，“设备”拥有“变量”。

新建 I/O 变量的一般步骤：新建端口→新建设备→新建变量

5.1 IO 设备管理

选择工程管理器，“数据库”下的“I/O 变量”，鼠标点击右键，出现“新建端口”菜单，点击“新建端口”，出现端口属性对话框。

The screenshot shows a dialog box titled "端口属性" (Port Properties) with a close button in the top right corner. The dialog is divided into several sections:

- General Settings:**
 - 端口名称 (Port Name): 端口1
 - 端口类型 (Port Type): 串口 (Serial)
 - 设备厂家 (Device Manufacturer): MODBUS
 - 设备参数 (Device Parameters): ...
 - 设备类型 (Device Type): Modbus_RTU
- 串口参数 (Serial Port Parameters):**
 - 串口号 (Serial Port): COM1
 - 波特率 (Baud Rate): 9600
 - 校验位 (Parity): 无校验 (None)
 - 数据位 (Data Bits): 8
 - 停止位 (Stop Bits): 1
 - 超时时间 (Timeout): 500 ms
- 以太网参数 (Ethernet Parameters):**
 - IP地址 (IP Address): 192 . 1 . 0 . 3
 - 端口号 (Port Number): 0

At the bottom of the dialog, there are two buttons: "确定" (OK) and "取消" (Cancel).

端口名称具有唯一性，不可以重复。

根据实际工程设备需求，可以选择设备类型，如“ModbusRTU”，选择端口类型为“串口”，设置好串口参数，点击“确定”即完成端口的定义。

注意：设备类型 若默认显示为所需型号，也须点下拉菜单重新选择。

定义完成后会在“I/O 变量”下面生成“端口 1”，右键点击“端口 1”，出现如下子菜单项。



点击新建设备，出现如下对话框。



根据实际要求，修改设备名称和地址。

点击确定后在“端口 1”下出现“设备 1”。



图 4.4 设备

假如工程有 2 个串口，第一个串口有 2 个“ModbusRTU”设备，第二个串口下有 3 个“ModbusRTU”设备，则可以建立“端口 1”，“端口 2”，“端口 1”下有 3 个设备，“端口 2”下有 2 个设备。如下图所示。



5.2 变量定义

选择相应的设备，然后右键点击右侧的变量显示区，出现如下子菜单。



有两种定义变量的方式：

- 1.如果变量寄存器地址是分散的，不连续的，可以点击“新建变量”来创建单个变量。
- 2.如果变量寄存器地址是连续的，可以点击“批量增加”来创建多个变量。

变量作为一个对象，其属性分为基本属性、存盘属性、报警属性、量程变换。

基本属性	定义
名称	变量在整个数据库中的唯一标志，不可重复。只能以字母开头。
描述	对变量名称的中文注释。
寄存器类型	不同的通讯规约，寄存器类型不一样，代表读取 PLC 的数据类型。
寄存器地址	变量对应的 PLC 中地址。
数据类型	变量对应的数据格式。

最小值	变量的最小值。
最大值	变量的最大值。
小数位数	变量值的小数有效位数。
读写属性	变量是否只读。
设值报警	设值时是否报警。

存盘属性有不存盘、定时存盘、存盘周期、数据变化存盘、存盘精度等。主要针对于模拟量输入信号。

不存盘：不保存历史数据。

定时存盘：按照规定的间隔时间存盘。（存盘数据每 5 分钟一次性写入 Nandflash）

数据变化存盘：根据设定的精度，达到一定的变化后存盘。

The screenshot shows a software interface for configuring variable properties. The title bar is '变量属性'. Below it are four tabs: '基本属性', '存盘属性', '报警属性', and '量程变换'. The '存盘属性' tab is active. It contains three radio button options: '不存盘' (selected), '定时存盘', and '数据变化存盘'. Next to '定时存盘' is a text label '存盘周期' followed by a numeric input field containing '0' and the unit '秒'. Next to '数据变化存盘' is a text label '存盘精度' followed by a numeric input field containing '0' and the unit '%'. The '不存盘' option has a small dashed box around it.

报警属性有模拟量报警设置和开关量报警设置。

报警等级有一般、严重、紧急三种。

变量属性

基本属性 | 存盘属性 | **报警属性** | 量程变换

模拟量报警设置

报警动作限值 报警等级

下下限报警 0 一般 越限死区 0

下限报警 0 一般

上限报警 0 一般

上上限报警 0 一般

大偏差报警 0 一般 偏差死区 0 目标值 0

小偏差报警 0 一般

变化率报警 0 一般 时间单位 秒

开关量报警设置

报警类型 不报警

0->1报警动作, 1->0报警复归 报警等级 一般

1->0报警动作, 0->1报警复归

变位报警 0状态描述 1状态描述

确定 取消

量程变换属性有进行量程变化、原始最小值、原始最大值等。

主要参数说明如下：

量程变换：针对模拟量信号进行线性变换用。

原始最小值：PLC 采集到的信号最小值。

原始最大值：PLC 采集到的信号最大值。

根据变量的实际情况，选择或者输入对应的属性，完成后点击“确认”即可完成变量定义。

变量属性

基本属性 | 存盘属性 | 报警属性 | **量程变换**

进行量程变换

原始最小值 0 原始最大值 0

批量增加变量，如下图操作。



序号	变量名	描述	寄存器类型	寄存器地址	数据类型	访问属性
1	I1S1		B-CX	1	BOOL型	只读
2	I1S2		B-CX	2	DOCC型	只读
3	I1S3		B-CX	3	DOCC型	只读
4	I1S4		B-CX	4	DOCC型	只读
5	I1S5		B-CX	5	DOCC型	只读
6	I1S6		B-CX	6	DOCC型	只读
7	I1S7		B-CX	7	DOCC型	只读
8	I1S8		B-CX	8	DOCC型	只读
9	I1S9		B-CX	9	DOCC型	只读
10	I1S10		B-CX	10	DOCC型	只读
11	I1S11		B-CX	11	DOCC型	只读
12	I1S12		B-CX	12	DOCC型	只读
13	I1S13		B-CX	13	DOCC型	只读
14	I1S14		B-CX	14	DOCC型	只读
15	I1S15		B-CX	15	DOCC型	只读
16	I1S16		B-CX	16	DOCC型	只读
17	I1S17		B-CX	17	DOCC型	只读
18	I1S18		B-CX	18	DOCC型	只读
19	I1S19		B-CX	19	DOCC型	只读
20	I1S20		B-CX	20	DOCC型	只读

5.3 变量类型

变量的基本类型共有三类：I/O 变量、内存变量、系统变量。

IO 变量 是指可与外部数据采集程序直接进行数据交换的变量，如下位机数据采集设备（如 PLC、仪表等）或其它应用程序（如 DDE、OPC 服务器等）。这种数据交换是双向的、动态的，就是说：在“CORWARE”系统运行过程中，每当 I/O 变量的值改变时，该值就会自动写入下位机或其它应用程序；每当下位机或应用程序中的值改变时，“CORWARE”系统中的变量值也会自动更新。所以，那些从下位机采集来的数据、发送给下位机的指令，比如“反应罐液位”、“电源开关”等变量，都需要设置成“I/O 变量”。

内存变量 是指那些不需要和其它应用程序交换数据、也不需要从下位机得到数据、只在“CORWARE”内需要的变量，比如计算过程的中间变量，就可以设置成“内存变量”。

系统变量 是指 CORWARE 系统定义的变量，如时、分、秒等，方便用户直接引用。

5.4 系统变量

定义如下：

\$YearNow, 短整型, 只读, 读取计算机系统内部的当前时间：“年”（1111~9999）。

\$MonthNow, 短整型, 只读, 读取计算机系统内部的当前时间：“月”（1~12）。

\$DayNow, 短整型, 只读, 读取计算机系统内部的当前时间：“日”（1~31）。

\$HourNow, 短整型, 只读, 读取计算机系统内部的当前时间：“小时”（0~24）。

\$MinuteNow, 短整型, 只读, 读取计算机系统内部的当前时间：“分钟”（0~59）。

\$SecondNow, 短整型, 只读, 读取计算机系统内部的当前时间：“秒数”（0~59）。

\$Week, 短整型, 只读, 读取 Wince 系统内部的当前星期几：“星期”（1~7），注意 1~7 对应于星期日~星期六，而不是星期一~星期天，如果要显示当前日期是星期几，切记要进行转换。

\$NewAlarmCount, 短整型, 只读, 新报警数目。

\$UserName, 字符串型, 只读, 当前用户名。

\$UserLevel, 短整型, 只读, 当前用户权限。

\$BacklightState, BOOL 型, 只读, 当前背光状态。

\$YearStart, 短整型, 只读, 系统启动时计算机系统内部的时间：“年”（1111~9999）。

\$MonthStart, 短整型, 只读, 系统启动时计算机系统内部的时间：“月”（1~12）。

\$DayStart, 短整型, 只读, 系统启动时计算机系统内部的时间：“日”（1~31）。

\$HourStart, 短整型, 只读, 系统启动时计算机系统内部的时间：“小时”（0~24）。

\$MinuteStart, 短整型, 只读, 系统启动时计算机系统内部的时间：“分钟”（0~59）。

\$SecondStart, 短整型, 只读, 系统启动时计算机系统内部的时间：“秒数”（0~59）。

\$U 盘状态, 短整型, 只读, 暂不支持。

\$SD 卡状态, 短整型, 只读, 暂不支持。

\$State_端口 1 设备 1, BOOL 型, 只读, 端口 1 下设备 1 通讯状态。

5.5 变量的数据类型

CORWARE 中变量的数据类型与一般程序设计语言中的变量比较类似，主要有以下几种：

- 短整型

类似一般程序设计语言中的无符号短整数型变量，用于表示无符号的短整型数据，取值范围 0~65535。

- 有符号短整型

类似一般程序设计语言中的有符号短整数型变量，用于表示带符号的短整型数据，取值范围-32768~32767。

- 长整型

类似一般程序设计语言中的无符号长整数型变量，用于表示无符号的长整型数据，取值范围 0~2⁶⁴。

- 有符号长整型

类似一般程序设计语言中的有符号长整数型变量，用于表示带符号的长整型数据，取值范围-2147483648~2147483647。

- 单精度实型变量

类似一般程序设计语言中的浮点型变量，用于表示浮点（float）型数据，取值范围±3.40282 * E+38，有效值 7 位。

- 双精度实型变量

类似一般程序设计语言中的浮点型变量，用于表示双精度浮点（double float）型数据，取值范围±1.7E*E+308，有效值 15 位。

- 字符串型变量

类似一般程序设计语言中的字符串变量，可用于记录一些有特定含义的字符串，如名称，密码等，该类型变量可以进行比较运算和赋值运算。

- 离散变量

类似一般程序设计语言中的布尔（BOOL）变量，只有 0，1 两种取值，用于表示一些开关量。

第六章 画面动态连接

工程人员在 CORWARE 开发系统中制作的画面都是静态的，那么它们如何才能反映工业现场的状况呢？这就需要通过实时数据库，因为只有数据库中的变量才是与现场状况同步变化的。数据库变量的变化又如何导致画面的动画效果呢？通过“动态连接”——所谓“动态连接”就是建立画面的图元与数据库变量的对应关系。这样，工业现场的数据，比如温度、液面高度等，当它们发生变化时，通过 I/O 接口，将引起实时数据库中变量的变化，如果设计者曾经定义了一个画面图元——比如指针——与这个变量相关，我们将会看到指针在同步偏转。

动态连接的引入是设计人机接口的一次突破，它把工程人员从重复的图形编程中解放出来，为工程人员提供了标准的工业控制图形界面，并且由可编程的命令语言连接来增强图形

界面的功能。图形对象与变量之间有丰富的连接类型，给工程人员设计图形界面提供了极大的方便。CORWARE 系统还为部分动态连接的图形对象设置了访问权限，这对于保障系统的安全具有重要的意义。

图形对象可以按动态连接的要求改变颜色、尺寸、位置、填充百分数等，一个图形对象又可以同时定义多个连接。把这些动态连接组合起来，应用程序将呈现出令人难以想象的图形动画效果。

6.1 动态连接属性框

给图形对象定义动态连接是在“动态属性”对话框中进行的。在 CORWARE 开发系统中选中图形对象，左键点击“属性框”，拉动滚动条，可以找到进行定义的动态属性。

注意：

对不同类型的图形对象的动态属性大致相同。但是对于特定属性对象，有些是灰色的，表明此动态连接属性不适应于该图形对象，或者该图形对象定义了与此动态连接不兼容的其它动态连接。

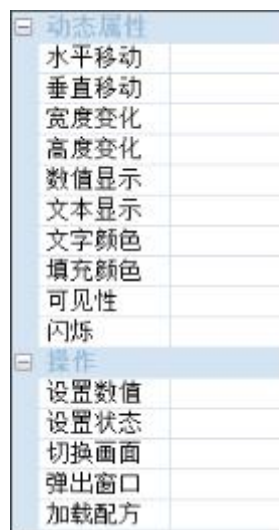


图 6.1 属性框

下面分组介绍所有的动态连接种类。

输出：文本图形对象能定义数值和字符串输出连接中的某一种。这种连接用来在画面上输出文本图形对象的连接表达式的值。运行时文本字符串将被连接表达式的值所替换，输出的字符串的大小、字体和文本对象相同。图符对象可以定义图符输出连接。按动按钮弹出相应的输出连接对话框。

颜色变化：共有三种连接（线条颜色、填充颜色、文本颜色），它们规定了图形对象的颜色、线型、填充类型等属性如何随变量或连接表达式的值变化而变化。单击按钮弹出相应的连接对话框。线类型的图形对象可定义线属性连接，填充形状的图形对象可定义线属性、填充属性连接，文本对象可定义文本色连接。

位置大小变化：这四种连接（水平移动、垂直移动、宽度、高度）规定了图形对象如何随变量值的变化而改变位置或大小。不是所有的图形对象都能定义这四种连接。单击按钮弹出相应的连接对话框。

其它：绝大多数图形对象都可以定义闪烁、可见性两种连接，这是两种规定图形对象可见性的连接。按动按钮弹出相应连接对话框。

动作：绝大多数图形对象都可以定义为五种动作中的一种。设置数值和设置状态，会弹出输入对话框，可以从键盘键入数据以改变数据库中变量的值。切换画面可以打开其它的画面。弹出窗口可以弹出小的自定义高度和宽度的窗口。加载配方可以加载需要的配方。对于动作，可以设置安全区，需要相应级别的用户登录后才可以输入。

6.2 动态连接表达式

在“动态属性”对话框中，单击任一种连接方式，将会弹出设置对话框，除可以直接连接变量值外，还可以连接简单的数学表达式。

6.2.1 ^ 运算符

计算数的指数次方。

6.2.2 * 运算符

两个数相乘。

如果需要将变量 AA 的值放大 10 倍，则输入 AA*10。

6.2.3 / 运算符

两个数值相除并返回以浮点数表示的结果。

如果需要将变量 AA 的值缩小 10 倍，则输入 AA/10。

6.2.4 + 运算符

计算两个数之和。

6.2.5 - 运算符

计算两个数值的差或表示数值表达式的负值。

6.3 动态连接详解

在“动态属性”对话框中，单击任一种连接方式，将会弹出设置对话框，本节详细解释各种动态连接的设置。

6.3.1 线条颜色

选中图元，在右侧【属性框】【动态属性】，单击“线条颜色”按钮，弹出连接对话框。线条颜色连接是使被连接对象的边框或线的颜色随连接表达式的值而改变。定义这类连接需要同时定义阈值和对应的线属性。利用连接表达式的多样性，可以构造出许多很有用的连接。

例如：可以用线条颜色表示离散变量 EXAM 的报警状态，只须在连接表达式中输入 EXAM，然后把下面的两个属性颜色对应的值改为 0（绿色），1（红色）即可。软件在运行时，当警报发生时（EXAM=1），线就由绿色变成了红色；当警报解除后，线又变为绿色。



图 6.2 线条颜色变化对话框

6.3.2 填充颜色

选中图元，在右侧【属性框】【动态属性】，单击“填充颜色”按钮，弹出连接对话框。填充颜色连接使图形对象的填充颜色随连接表达式的值而改变，通过定义一些分断点（包括阈值和对应填充属性），使图形对象的填充颜色在一段数值内为指定值。

本例为封闭图形对象定义填充属性连接，阈值为 0 时填充属性为红色，阈值为 1 时为绿色。



图 6.3 填充颜色变化对话框

6.3.3 文字颜色

选中图元，在右侧【属性框】【动态属性】，单击“文字颜色”按钮，弹出连接对话框。文本颜色连接是使文本对象的颜色随连接表达式的值而改变，通过定义一些阈值及对应颜色，使文本颜色随变量值变化而变化。阈值是 0，文本色为黑色，阈值为 1，文本色为黑色。



图 6.4 文本颜色变化对话框

6.3.4 水平移动

选中图元，在右侧【属性框】【动态属性】，单击“水平移动”按钮，弹出连接对话框。水平移动连接是使被连接对象在画面中随连接表达式值的改变而水平移动。移动距离以像素为单位，以被连接对象在画面制作系统中的原始位置为参考基准的。水平移动连接常用来表示图形对象实际的水平运动。

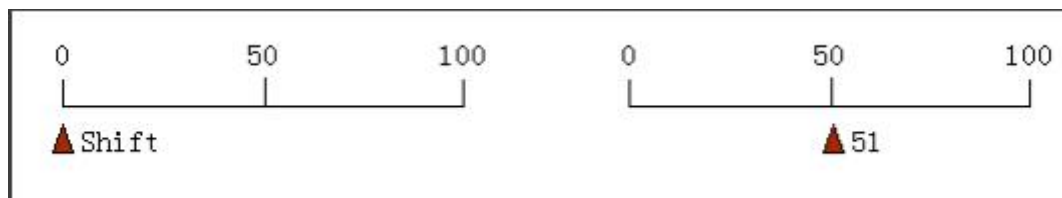


图 6.5 水平移动示例

本例中建立一个指示器，在画面上画一三角形（将其设置“水平移动”动画连接属性），以表示 shift 量的实际大小。



图 6.6 位置变化对话框

6.3.5 垂直移动

选中图元，在右侧【属性框】【动态属性】，单击“垂直移动”按钮，弹出连接对话框。垂直移动连接是使被连接对象在画面中随连接表达式值的改变而垂直移动。移动距离以像素为单位，以被连接对象在画面制作系统中的原始位置为参考基准的。垂直移动连接常用来表示图形对象实际的垂直运动。



图 6.7 位置变化对话框

6.3.6 宽度变化

选中图元，在右侧【属性框】【动态属性】，单击“宽度变化”按钮，弹出连接对话框。宽度变化连接是使被连接对象的宽度随连接表达式的值而变化。例中建立一个进度条，用一矩形（将其设置“缩放连接”动画连接属性），以反映变量“进度”的变化



6.3.7 高度变化

选中图元，在右侧【属性框】【动态属性】，单击“高度变化”按钮，弹出连接对话框。高度变化连接是使被连接对象的高度随连接表达式的值而变化。例中建立一个温度计，用一矩形表示水银柱（将其设置“缩放连接”动画连接属性），以反映变量“温度”的变化

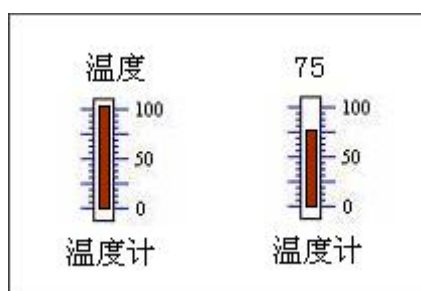


图 6.8 高度变化示例



图 6.9 大小变化对话框

6.3.8 数值显示

选中图元，在右侧【属性框】【动态属性】，单击“数值显示”按钮，弹出连接对话框。

数值显示是使文本对象的内容在程序运行时被连接表达式的值所取代。

显示小数时，需要填写小数位数，整数位数可以写 0。

注：小数位只可以写入 0 到 4 的整数。



6.3.9 文本显示

选中图元，在右侧【属性框】【动态属性】，单击“文本显示”按钮，弹出连接对话框。字符串连接是程序运行时使画面中文本对象的内容随连接表达式的值而改变。通过定义一些阈值（包括值和对应字符串），使文本显示在特定数值时为指定字符串。字符串连接除上述连接外，还可以直接连接字符串变量，使画面中文本对象的内容在程序运行时被数据库中的某个字符串变量的值所取代。

例：变量“B3”值为 0 时，文本显示为“运行正常”，值为 1，文本显示为“运行故障”。

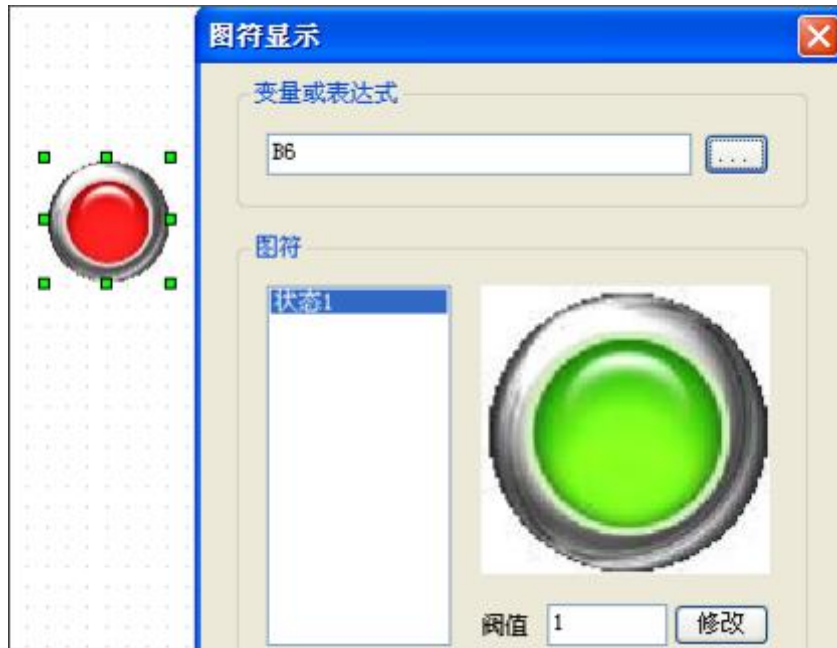


图 6.13 显示字符串对话框

6.3.10 图符显示

选中图元，在右侧【属性框】【动态属性】，单击“图符显示”按钮，弹出连接对话框。

例：变量“B6”值为0时指示灯为红色，变量“B6”值为1时指示灯为绿色。



6.3.11 可见性

选中图元，在右侧【属性框】【动态属性】，单击“可见性”按钮，弹出连接对话框。



图 6.14 可见性对话框

6.3.12 闪烁

选中图元，在右侧【属性框】【动态属性】，单击“闪烁”按钮，弹出连接对话框。



6.3.13 设置数值

选中图元，在右侧【属性框】【操作】，单击“设置数值”按钮，弹出连接对话框。设置数值是使被连接对象在运行时为触敏对象，单击此对象或按下指定热键将弹出输入值对话框，用户在对话框中可以输入连接变量的新值，以改变数据库中某个模拟型变量或非位变量的值。



图 6.16 设置数值对话框

6.3.14 设置状态

选中图元，在右侧【属性框】【操作】，单击“设置状态”按钮，弹出连接对话框。设置状态是使被连接对象在运行时为触敏对象，单击此对象根据预先设置的方式改变数据库中某个离散类型变量的值。

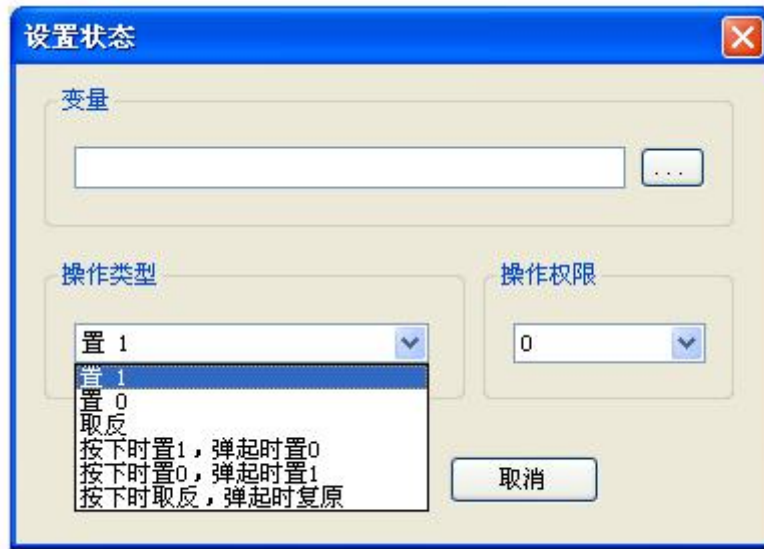


图 6.17 设置状态对话框

6.3.15 切换画面

选中图元，在右侧【属性框】【操作】，单击“切换画面”按钮，弹出连接对话框。切换画面是运行时根据预先设置的画面名称打开画面。



图 6.18 切换画面对话框

6.3.16 弹出窗口

选中图元，在右侧【属性框】【操作】，单击“弹出窗口”按钮，弹出连接对话框。弹出窗口是根据预先设置的窗口名称打开窗口，设置“窗口位置”可以设定窗口打开后的位置坐标。注：窗口的大小设置要小于画面的尺寸。



图 6.19 弹出窗口对话框

6.3.17 加载配方

选中图元，在右侧【属性框】【操作】，单击“加载配方”按钮，弹出连接对话框。加载配方是使被连接对象在运行时根据预先设置的配方，显示配方内容并可以操作配方。



加载配方

当前配方

甜面包

编辑配方

当前配方内容

变量名	值
L糖	80
L盐	10
L面粉	80
L水	30
L蜂蜜	10

第七章 曲线显示历史数据显示

在实际生产过程中，对实时数据、历史数据的查看、分析是不可缺少的工作，但对大量数据仅做定量的分析还远远不够，必须根据大量的数据信息，绘制出趋势曲线，从趋势曲线的变化中发现数据的变化规律。因此，趋势曲线处理在工控系统中是一个非常重要的部分。

CORWARE 为用户提供强大的趋势曲线功能。通过众多功能各异的曲线构件，包括历史曲线、实时曲线，用户能够组态出各种类型的趋势曲线，从而满足不同工程项目的各种需求。

CORWARE 共提供了四种用于趋势曲线绘制的构件，分别是：实时曲线、历史曲线、计划曲线、XY 曲线。每种曲线构件的功能各不相同：

实时曲线：

实时曲线是在系统运行时，从实时数据库中读取数据，同时，以时间为 X 轴，数据值为 Y 轴进行曲线绘制。

历史曲线：

历史曲线是将历史存盘数据从数据库中读出，以时间为 X 轴，数据值为 Y 轴进行曲线绘制。同时，历史曲线也可以实现实时刷新的效果。历史曲线主要用于事后查看数据分布和状态变化趋势以及总结信号变化规律。

计划曲线：

计划曲线是将生产过程中的数据预先以曲线的方式设置，在启动生产后，可以查看计划与实际是否一致，便于生产人员根据数据进行对照，分析后修改完善程序，达到最理想的生产状态。

XY 曲线：

XY 曲线是提取寄存器中的数据，X 轴均匀分布，数据值为 Y 轴进行曲线绘制。

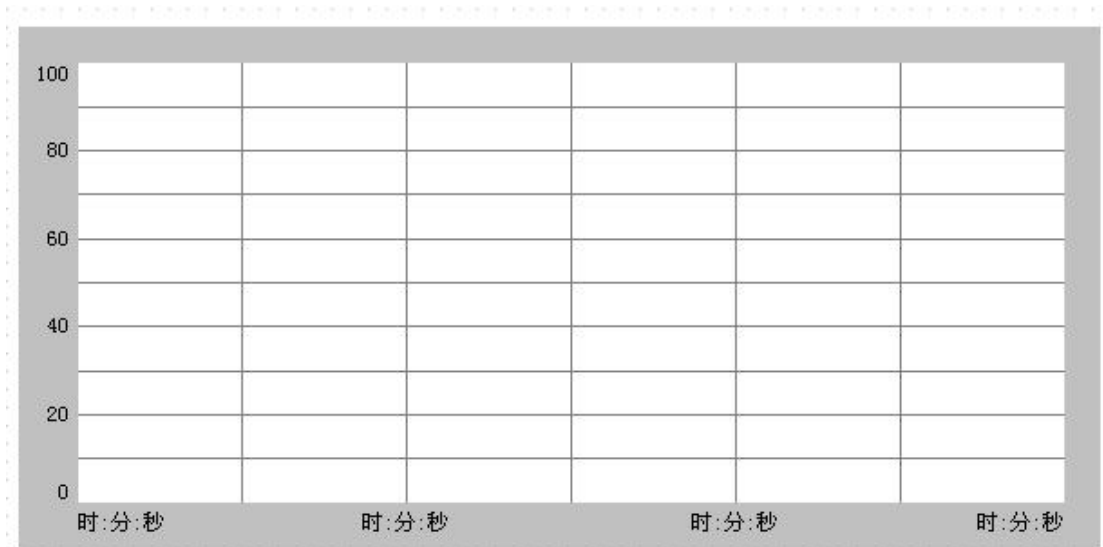
7.1 实时曲线

实时曲线具有以下特点：

- 1、单个实时曲线图元最多可以显示 8 条曲线。
- 2、可以设置每条曲线的颜色。
- 3、可以设置背景、边框、坐标、网格颜色。
- 4、可以设置坐标参数。

7.1.1 创建实时曲线图元

选择菜单“图元”下的“实时曲线”菜单，鼠标箭头变为小“十”字型，在画面上选择控件的左上角，按下鼠标左键并拖动，画面上显示出一个矩形框，该矩形框为创建后的曲线的外框。当达到所需大小时，松开鼠标左键，则实时曲线控件创建成功，画面上显示出该曲线，如图所示。



7.1.2 实时曲线属性设置

实时曲线图元创建完成后，在属性框中选择“设置”命令，选择“曲线设置”，弹出实时曲线图形的属性设置对话框，有 3 个属性框，如图所示。



基本属性	释义
窗口背景色	设置窗口背景颜色，7.1.1 实时曲线中灰色部分
曲线背景色	设置曲线背景颜色，7.1.1 实时曲线中白色部分
窗口边框	设置窗口边框无、凹下、凸起
曲线边框	设置曲线边框无、凹下、凸起
曲线名称显示类型	设置曲线名称显示位置，可以设置为不显示、在曲线上侧显示、在曲线下侧显示、在曲线右侧显示
曲线名称	设置曲线名称显示位置、曲线名称的颜色和字体类型



标注属性	释义
时间轴时间长度	设置显示时间的长短，可以设为秒、分、时、天。
时间轴采样间隔	设置采样值的时间间隔，可以设为分钟或秒钟。
时间轴颜色	设置时间字体的颜色。
时间轴字体	设置时间轴文字字体类型。
时间轴标注间隔	设置时间轴标注间隔，间隔根据 X 主划线确认。
时间轴时间格式	设置时间轴时间格式。
数值轴最大值	设置曲线左侧 Y 轴数值的最大值。
数值轴最小值	设置曲线左侧 Y 轴的最小值。
数值轴标注间隔	设置数值轴标注间隔，间隔根据 Y 主划线确认。
数值轴小数位数	设置小数有效位数。
数值轴颜色	设置数值轴字体的颜色。
数值轴字体	设置数值轴文字字体类型。



曲线属性中设置如下：共 4 条曲线可以设置。

曲线属性	释义
变量名	设置曲线关联的变量名称
颜色	设置曲线显示颜色
线宽	设置曲线显示线宽

7.2 历史曲线

历史曲线具有以下特点：

- 1、单个历史曲线图元最多可以显示 8 条曲线。
- 2、可以设置每条曲线的颜色。
- 3、可以设置背景、边框、坐标、网格颜色。
- 4、可以设置坐标参数。

7.2.1 创建历史曲线图元

选择菜单“图元”下的“历史曲线”菜单，鼠标箭头变为小“十”字型，在画面上选择控件的左上角，按下鼠标左键并拖动，画面上显示出一个矩形框，该矩形框为创建后的曲线

的外框。当达到所需大小时，松开鼠标左键，则历史曲线控件创建成功，画面上显示出该曲线，如图所示。



7.2.2 历史曲线属性设置

历史曲线图元创建完成后，在右侧属性框中选择“设置”命令，选择“曲线设置”，弹出历史曲线图形的属性设置对话框，有3个属性框，参数意义与实时曲线相同。

基本属性	标注属性	曲线属性
时间轴		
时间长度	<input type="text" value="1"/>	分
采样间隔	<input type="text" value="1"/>	秒
颜色	<input type="color" value="black"/>	字体 <input type="button" value="..."/>
标注间隔	<input type="text" value="2"/>	
时间格式	<input type="text" value="时:分:秒"/>	
数值轴		
最大值	<input type="text" value="100"/>	最小值 <input type="text" value="0"/>
标注间隔	<input type="text" value="2"/>	小数位数 <input type="text" value="0"/>
颜色	<input type="color" value="black"/>	字体 <input type="button" value="..."/>

基本属性	标注属性	曲线属性
	变量名	颜色 线宽
曲线1	<input type="text"/>	<input type="button" value="..."/> <input type="color" value="black"/> 1
曲线2	<input type="text"/>	<input type="button" value="..."/> <input type="color" value="black"/> 1
曲线3	<input type="text"/>	<input type="button" value="..."/> <input type="color" value="black"/> 1
曲线4	<input type="text"/>	<input type="button" value="..."/> <input type="color" value="black"/> 1

7.3 计划曲线

计划曲线具有以下特点：

- 1、可以设置分段点配方。
- 2、可以设置每条曲线的颜色。
- 3、可以设置背景、边框、坐标、网格颜色。
- 4、可以设置坐标参数。

7.3.1 创建计划曲线图元

选择菜单“图元”下的“计划曲线”菜单，鼠标箭头变为小“十”字型，在画面上选择控件的左上角，按下鼠标左键并拖动，画面上显示出一个矩形框，该矩形框为创建后的曲线的外框。当达到所需大小时，松开鼠标左键，则计划曲线控件创建成功，画面上显示出该曲线，如图所示。



计划曲线图元

7.3.2 计划曲线属性设置

计划曲线图元创建完成后，在右侧属性框中选择“设置”命令，选择“曲线设置”，弹出计划曲线图形的属性设置对话框，如图所示。属性参数与其它曲线类似。





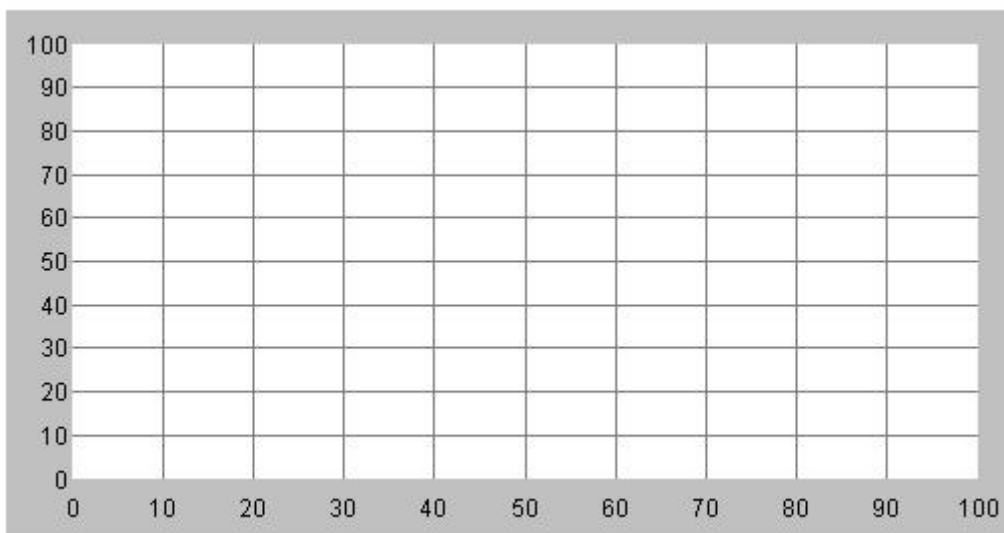
7.4 XY 曲线

XY 曲线具有以下特点：

- 1、通过脚本描点显示 XY 曲线。
- 2、可以设置每条曲线的颜色。
- 3、可以设置背景、边框、坐标、网格颜色。
- 4、可以设置坐标参数。

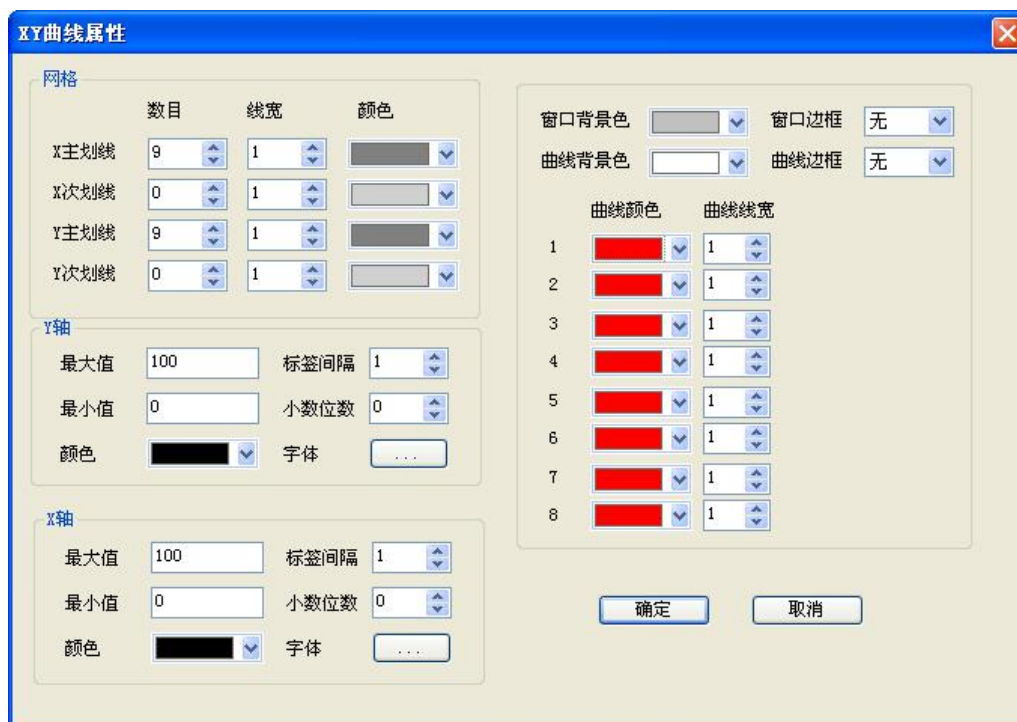
7.4.1 创建实时曲线图元

选择菜单“图元”下的“XY 曲线”菜单，鼠标箭头变为小“十”字型，在画面上选择控件的左上角，按下鼠标左键并拖动，画面上显示出一个矩形框，该矩形框为创建后的曲线的外框。当达到所需大小时，松开鼠标左键，则实时曲线控件创建成功，画面上显示出该曲线，如图所示。



7.4.2 XY 曲线属性设置

实时曲线图元创建完成后，在属性框中选择“设置”命令，选择“曲线设置”，弹出实时曲线图形的属性设置对话框，如图所示。



曲线属性中属性如下：

“窗口背景色”用来设置窗口背景颜色。

- “曲线背景色” 用来设置曲线背景颜色。
- “窗口边框” 用来设置窗口边框无、凹下、凸起。
- “曲线边框” 用来设置曲线边框无、凹下、凸起。
- “曲线颜色” 用来设置曲线颜色。
- “曲线线宽” 用来设置曲线宽度。
- “X 轴最大值”， 用来设置 X 轴的最大值。
- “X 轴最小值”， 用来设置 X 轴的最小值。
- “X 轴标签间隔” 用来设置 X 轴标签间隔。
- “X 轴小数位数” 用来设置 X 轴小数有效位数。
- “X 轴颜色” 用来设置 X 轴的颜色。
- “X 轴字体” 用来设置 X 轴文字字体。
- “Y 轴最大值”， 用来设置 Y 轴的最大值。
- “Y 轴最小值”， 用来设置 Y 轴的最小值。
- “Y 轴标签间隔” 用来设置 Y 轴标签间隔。
- “Y 轴小数位数” 用来设置 Y 轴小数有效位数。
- “Y 轴颜色” 用来设置 Y 轴的颜色。
- “Y 轴字体” 用来设置 Y 轴文字字体。

7.4.3 脚本

函数 SetTrendMAXPointNum 设置 X 轴总点数

函数 SetTrendXPointValue 设置 X 轴数值

函数 SetTrendYPointValue 设置 Y 轴数值

脚本样例：

```
document.SetTrendMAXPointNum "Obj1350",64
For i=0 To 63
x1=i
y1=document.GetPointValue("W4X"&CStr(450+i))
y2=document.GetPointValue("W4X"&CStr(520+i))
y3=document.GetPointValue("W4X"&CStr(590+i))
document.SetTrendXPointValue "Obj1350",i,x1,x1,x1,0,0,0,0,0
document.SetTrendYPointValue "Obj1350",i,y1,y2,y3,0,0,0,0,0
```



7.5 历史数据表格

历史数据表格具有以下特点：

- 1、历史表格图元可以最多同时显示 30 个数据的历史数据。
- 2、可以设置每列数据的列名。
- 3、可以设置背景、边框、坐标、网格颜色。
- 4、可以修改每列的宽度、数据的显示格式。
- 5、可以以.csv 格式导出已经查询到的数据。

7.5.1 创建历史数据表格图元

选择菜单“图元”下的“历史数据”菜单，鼠标箭头变为小“十”字型，在画面上选择控件的左上角，按下鼠标左键并拖动，画面上显示出一个矩形框，该矩形框为创建后的表格的外框。当达到所需大小时，松开鼠标左键，则历史数据控件创建成功，画面上显示出该表格，如图所示。

第八章 报表系统

数据报表是反应生产过程中的数据、状态等，并对数据进行记录的一种重要形式。是生产过程必不可少的一部分。它既能反映系统实时的生产情况，也能对长期的生产过程进行统计、分析，使管理人员能够实时掌握和分析生产情况。

CORWARE 提供内嵌式报表系统，工程人员可以任意设置报表格式，对报表进行组态。为工程人员提供了丰富的报表函数，实现各种运算、数据转换、统计分析、报表打印等。既可以制作实时报表，也可以制作历史报表。

8.1 如何创建报表

8.1.1 新建报表

点击“工程管理器”，右键点击“日报表”或“月报表”或“年报表”，弹出“新建报表”菜单。点击“新建报表”，出现“报表属性”对话框。

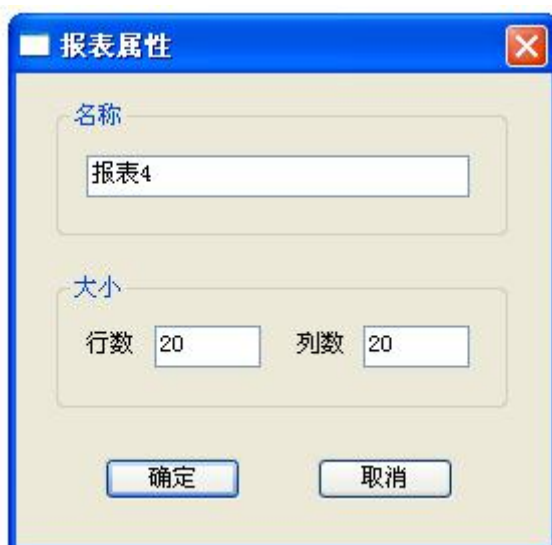


图 8.1 报表属性

在报表属性对话框中，名称为报表的名称，行数为报表的行数，列数为报表的列数。点击“确定”，报表创建成功。

8.1.2 报表单元格样式

行用数字“1、2、3…”表示，列用英文字母“A、B、C、D…”表示。单元格的名称定义为“列标+行号”，如“a1”，表示第一行第一列的单元格。列标使用时不区分大小写，如“A1”和“a1”都可以表示第一行第一列的单元格。

点击报表单元格，在右边的属性框中可以看到如下属性。



点击“格式”右边的列，出现格式属性。



点击“边框”右边的列，出现边框属性框。



8.2 加载报表和显示数据

创建完成报表后，可通过【菜单栏】【插入】【报表】或直接点击图元栏 ，在编辑框中拖出一个报表图元，然后可通过脚本加载和显示数据

8.2.1 加载报表

语法格式：`void ReportLoad(BSTR strObjName,BSTR strReportName)`。

`strObjName` 图元对象名称，`strReportName` 报表名称。

例程：要在图元“Obj1”中加载“报表1”报表，`document.ReportLoad "Obj1","报表1"`。

8.2.2 显示数据

语法格式：`void Report_SetCellText(BSTR strObjName,int nRow, int nCol, BSTR str)`。

`strObjName` 图元对象名称。

`nRow` 单元格行号，从 0 开始。

`nCol` 单元格列号，从 0 开始。

`str` 单元格内容，字符串类型。

例程：要在图元“Obj1”中第 1 行第 1 列显示数据“abc”，


```
document.Report_SetCellText " Obj1",1,1,"abc"
```

	abc			

第九章 配方管理

9.1 概述

9.1.1 什么是配方

什么是配方？在制造领域，配方是用来描述生产一件产品所用的不同配料之间的比例关系。配方是生产过程中一些变量对应的参数设定值的集合。例如，一个面包厂生产面包时有一个基本的配料配方，此配方列出所有要用来生产面包的配料成份表（如水，面粉，糖，鸡蛋，香油等）。另外，也列出所有可选配料成份表（如果酱，维生素，巧克力等），而这些可选配料成份可以被添加到基本配方中用以生产各种各样的面包。下表为某一面包厂生产面包时的配方：

	糖	盐	面粉	水	蜂蜜
甜面包	80	10	80	30	10
低糖面包	30	5	80	30	0
无糖面包	10	5	80	30	0

表 9.1 面包配方表

9.1.2 配方管理

CORWARE 提供的配方管理由配方管理器。配方管理器打开后，弹出对话框，用于创建和维护配方模板文件。所有配方都在配方文件中定义和存储，每一个配方文件以扩展名为.rcp 的文件格式存储。

右键点击“配方管理”，出现“新建配方组”。



图 9.1 配方管理

点击“新建配方组”出现配方编辑对话框。

配方编辑				
配方组名称:		面包配方		
	变量名	配方1	配方2	配方3
配方名		甜面包	低糖面包	无糖面包
变量1	L糖	80	30	10
变量2	L盐	10	5	5
变量3	L面粉	80	80	80
变量4	L水	30	30	30
变量5	L蜂蜜	10	0	0

图 9.2 配方编辑对话框

配方组和配方:

每个配方组就是一张表格，每个配方就是表格中的一行，而表格的每一列就是配方组的一个成员变量。

配方组名称:

配方组的名称应能够清楚反映配方的实际用途，例如面包配方组就是各种面包的配方。

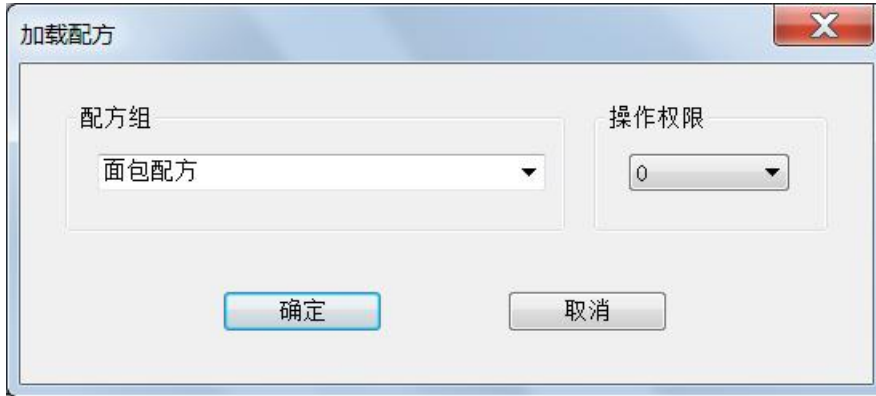
变量个数:

这里的变量个数就是配方组成员变量的数量，也就是配方中的原料总数。例如上表的配方就有 5 种原料，那么对应的配方组就应该有 5 个成员变量

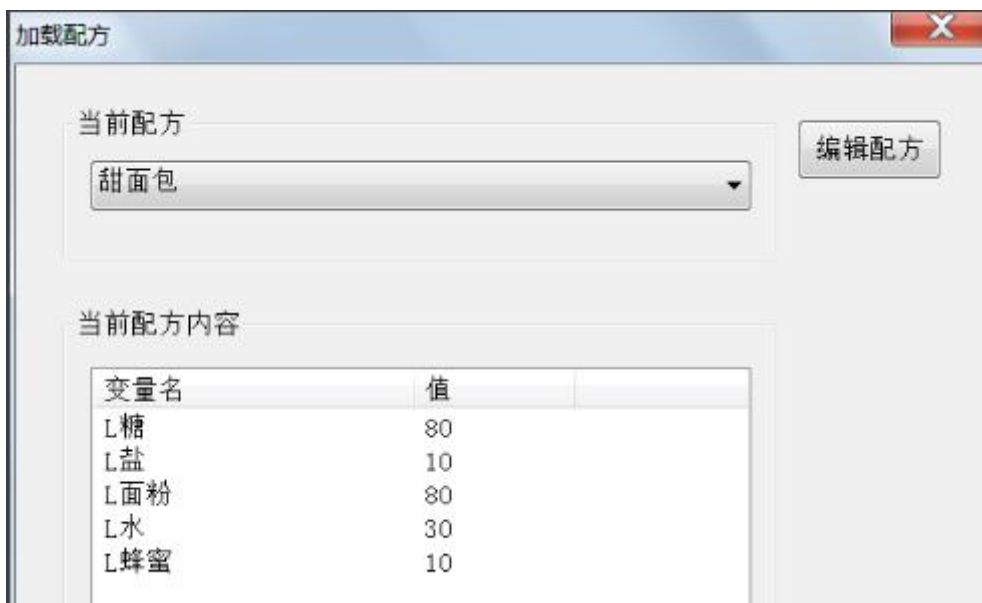
9.1.3 如何使用配方

A. CORWARE 在动态连接中提供“加载配方”功能，一个按钮便可以调用相应的配方文件。

1. 点击【属性框】【操作】【加载配方】右侧按钮弹出加载配方窗口，选择配方组。



2. 程序运行时点击按钮，弹出“加载配方”窗口，选择需要的配方，点击确定，即可将配方值加载到对应的变量中。



B. 利用脚本可直接加载配方：**RecipeLoad** “配方组名”，“配方名”。

新建一个按钮，在【属性框】【脚本】【弹起时执行】，输入以下脚本，运行时点击该按钮，可直接将甜面包的配方加载到对应的变量中。

RecipeLoad “面包配方”，“甜面包”

第十章 系统安全管理

安全保护是应用系统不可忽视的问题,对于可能有不同类型的用户共同使用的大型复杂应用,必须解决好授权与安全性的问题,系统必须能够依据用户的使用权限允许或禁止其对系统进行操作。CORWARE 提供一个强有力的先进的基于用户的安全管理系统。在 CORWARE 系统中,在开发系统里可以对工程进行加密。打开工程时只有输入密码正确时才能进入该工程的开发系统。上载工程时也只有输入密码正确时才能上载工程。对画面上的图形对象设置访问权限,同时给操作者分配访问优先级,运行时当操作者的优先级小于对象的访问优先级,该对象为不可访问,即要访问一个有权限设置的对象,要求先具有访问优先级,方能访问。CORWARE 以此来保障系统的安全运行。

10.1 开发系统安全管理

10.1.1 如何对工程进行加密

为了防止其他人员对工程进行修改,在 CORWARE 开发系统中可以分别对多个工程进行加密。当进入一个有密码的工程时,必须正确输入密码方可进入开发系统,否则不能打开该工程进行修改,从而实现了 CORWARE 开发系统的安全管理。

新建 CORWARE 工程，首次进入 CORWARE 工程管理器，系统默认没有密码，可直接进入 CORWARE 开发系统。如果要对该工程的开发系统进行加密，执行“工具”——>“工程加密”，弹出“工程加密”对话框，如图所示。



图 10.1 工程加密对话框

旧密码：工程的当前密码，密码长度不超过 17 个字节，密码可以是字母（区分字母大小写）、数字、其它符号等。

新密码：工程的新密码，密码长度不超过 17 个字节，密码可以是字母（区分字母大小写）、数字、其它符号等。

新密码确认：再次输入新密码确认。

单击取消按钮将取消对工程实施加密操作；单击确定按钮后，系统将对工程进行加密。

退出 CORWARE，每次在开发环境下打开该工程都会出现检查文件密码对话框，要求输入密码，如图所示。



图 10.2 密码验证对话框

密码输入正确后，将打开该工程。否则出现如图所示对话框。



图 10.3 密码错误对话框

10.1.2 如何去除工程加密

如果想取消对工程的加密，在打开该工程后，单击“工具\工程加密”，弹出“工程加密处理”对话框，输入旧密码，同时将新密码和新密码确认设为空，单击确定按钮，即可取消工程加密。

注意：

如果用户丢失工程密码，将无法打开 CORWARE 工程进行修改，请小心妥善保存密码！

10.1.3 如何设置工程上载密码

为了有效的保护知识产权，在 CORWARE 开发系统中可以对工程上载加上密码保护。未经授权的上载会被拒绝。打开工程后，点击“工具\工程设置\上下载设置”，在上载密码编辑框中输入密码，保存工程。



图 10.4 设置工程上载密码

10.2 运行系统安全管理

10.2.1 运行系统安全管理概述

为了保证运行系统的安全运行，对画面上的图形对象设置访问权限，同时给操作者分配访问优先级，当操作者的优先级小于对象的访问优先级时，该对象为不可访问，即要访问一个有权限设置的对象，要求先具有访问优先级，方能访问。操作者的操作优先级级别从 1~16，每个操作者和对象的操作优先级级别只有一个。

10.2.2 安全管理配置

10.2.2.1 优先级

采用分优先级的保护策略。系统将优先级从小到大定为 1 到 16，可以对用户、图形对象等设置不同的优先级。

可定义操作优先级的为五种动作（设置数字、设置状态、切换画面、弹出窗口、加载配方）。

10.2.2.2 如何配置用户

工程管理器中，单击“用户管理”，在右侧的内容区，点击右键，出现菜单。



图 10.5 用户管理

单击“新建用户”，出现用户属性框。



图 10.6 用户属性

10.2.3 运行时如何登录用户

在运行环境下，操作人员必须以自己的身份登录才能获得一定的操作权。

登录用户通过脚本语言进行，UserLogin (0)，如果未登录，点击后登录，如果已登录，则点击后退出登录。

10.2.4 与安全管理相关的系统变量和函数

在与安全管理有关的系统变量有两个：“\$UserName”和“\$UserLevel”。

“\$UserName”代表当前用户名，在程序运行时记录当前用户的名字。若没有用户登录或用户已退出登录，“\$UserName”为“未登录”。

“\$UserLevel”代表当前用户权限。在程序运行时记录着当前用户的访问权限。若没有用户登录或用户已退出登录，“\$UserLevel”为0。

注意：“\$UserName”为字符串变量，“\$UserLevel”为短整型变量。

第十一章 报警和事件系统

为保证工业现场安全生产，报警和事件的产生和记录是必不可少的。CORWARE 提供了强有力的报警和事件系统，并且操作方法简单。

11.1 关于报警和事件

报警是指当系统中某些量的值超过了所规定的界限时，系统自动产生相应警告信息，表明测量的值已经超限，提醒操作人员。如炼油厂的油品储罐，如果往罐中输油时，如果没有规定油位的上限，系统就产生不了报警，无法有效提醒操作人员，则有可能会造成“冒罐”，形成危险。有了报警，就可以提示操作人员注意。报警允许操作人员应答。

事件是指用户对系统的行为、动作。如修改了某个变量的值，用户的登录、注销等。事件不需要操作人员应答。

11.2 如何定义变量的报警属性

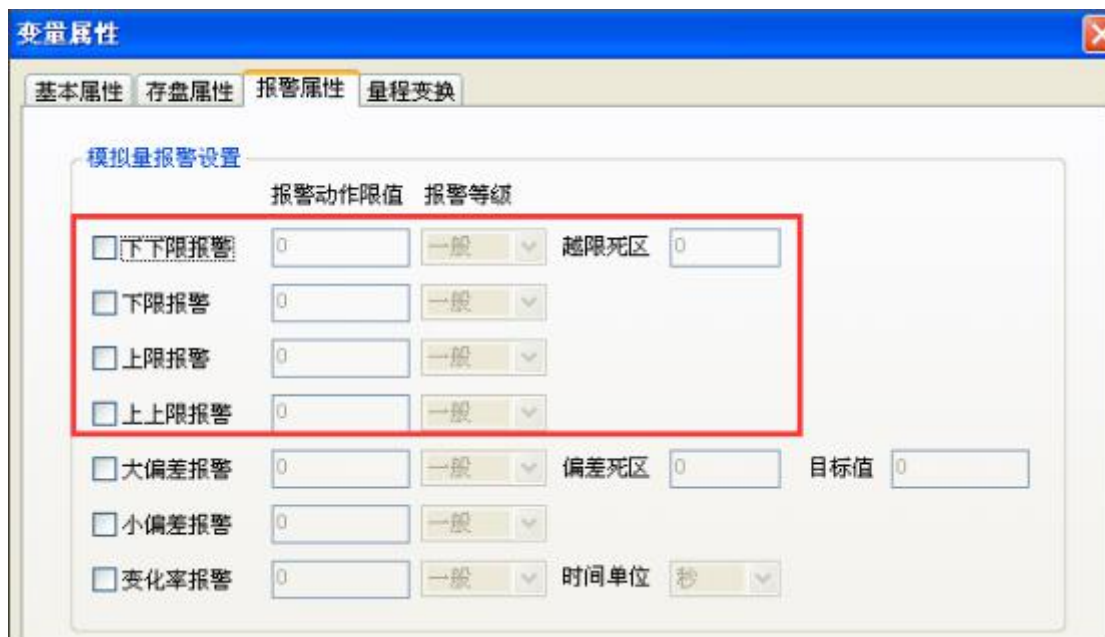
在使用报警功能前，必须先要对变量的报警属性进行定义。HDMS 的变量中模拟型（包括整型和实型）变量和离散型变量可以定义报警属性。

在数据库中双击变量名，弹出属性框，选择“报警属性”属性页，即可对各类报警属性进行设置。

11.2.1 模拟量变量的报警类型

模拟量主要是指整型变量和实型变量，主要是 IO 型的。模拟型变量的报警类型主要有三种：**越限报警**、**偏差报警**和**变化率报警**。

11.2.1.1 越限报警



模拟量的值在跨越规定的高低报警限时产生的报警。越限报警的报警限共有四个：低低限、低限、高限、高高限。其原理图如图所示。



在变量值发生变化时，如果跨越某一个限值，立即发生越限报警，某个时刻，对于一个变量，只可能越一种限，因此只产生一种越限报警，例如：如果变量的值超过高高限，就会产生高高限报警，而不会产生高限报警。另外，如果两次越限，就得看这两次越的限是否是同一种类型，如果是，就不再产生新报警，也不表示该报警已经恢复；如果不是，则先恢复原来的报警，再产生新报警。越限报警产生和恢复的算法为：

大于低低限时恢复低低限，小于等于低低限时产生报警；

大于低限时恢复低限，小于等于低限时报警产生报警；

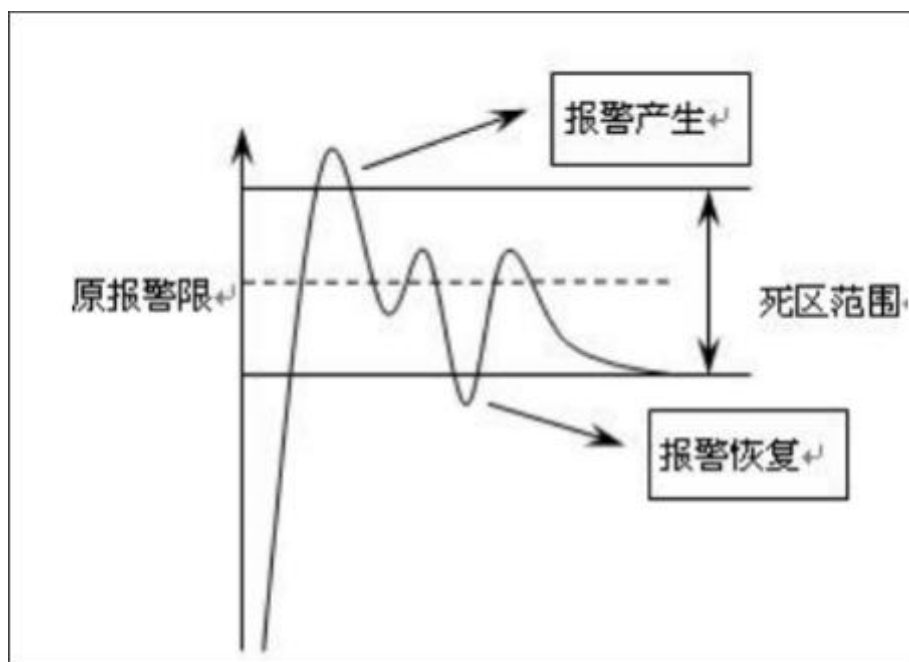
大于等于高限时报警，小于高限时恢复高限；

大于等于高高限时报警，小于高高限时恢复高高限；

限值报警列中选择要定义的越限类型，则后面的报警动作限值和报警等级变为有效。定义界限值时应该：最小值 \leq 低低限值 $<$ 低限 $<$ 高限 $<$ 高高限 \leq 最大值。

报警死区

越限报警死区的作用是为了防止变量值在报警限上下频繁波动时，产生许多不真实的报警，在原报警限上下增加一个报警限的阈值，使原报警限界线变为一条报警限带，当变量的值在报警限带范围内变化时，不会产生和恢复报警，而一旦超出该范围时，才产生报警信息。这样对消除波动信号的无效报警有积极的作用。报警死区的原理图如图所示。



11.2.1.2 偏差报警

模拟量的值相对目标值上下波动超过指定的变化范围时产生的报警。偏差报警可以分为小偏差和大偏差报警两种。当波动的数值超出大小偏差范围时，分别产生大偏差报警和小偏差报警，其原理如下图所示。

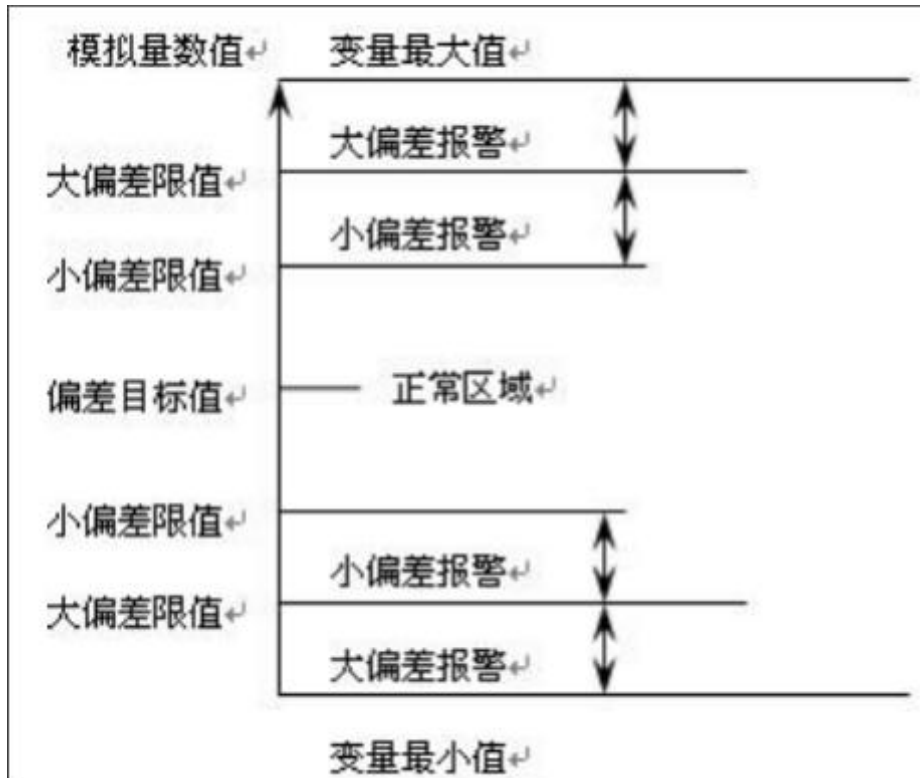


图 11.3 偏差报警原理

偏差报警限的计算方法为：

小偏差报警限=偏差目标值±定义的小偏差；

大偏差报警限=偏差目标值±定义的大偏差；

大于等于小偏差报警限时，产生小偏差报警；

大于等于大偏差报警限时，产生大偏差报警；

小于等于小偏差报警限时，产生小偏差报警；

小于等于大偏差报警限时，产生大偏差报警；

偏差报警在使用时可以按照需要定义一种偏差报警或两种都使用。

变量变化的过程中，如果跨越某个界限值，则立刻会产生报警，而同一时刻，不会产生两种类型的偏差报警。

例如：某一工序中要求压力在一定的范围内，不能太大，也不能太小，这是可以定义偏

差报警来确定压力的值是否在要求的范围内。

11.2.1.3 变化率报警

变化率报警是指模拟量的值在一段时间内产生的变化速度超过了指定的数值而产生的报警，即变量变化太快时产生的报警。系统运行过程中，每当变量发生一次变化，系统都会自动计算变量变化的速度，以确定是否产生报警。

变化率报警的类型以时间为单位分为三种：%x/秒、%x/分、%x/时。变化率报警的计算公式如下：

$$\left(\frac{(\text{变量的当前值} - \text{变量变化前的值}) \times 100}{(\text{变量本次变化的时间} - \text{变量上一次变化的时间}) \times (\text{变量的最大值} - \text{变量的最小值}) \times (\text{报警类型单位对应的值})} \right)$$

其中报警类型单位对应的值定义为：如果报警类型为秒，则该值为 1；如果报警类型为分，则该值为 60；如果报警类型为时，则该值为 3600。

取计算结果的整数部分的绝对值作为结果，若计算结果大于等于报警极限值，则立即产生报警。变化率小于报警极限值时，报警恢复。

11.2.2 开关量变量的报警类型

离散量有两种状态：1、0。离散型变量的报警有三种状态：

- 1、 1 状态报警：变量的值由 0 变为 1 时产生报警；
- 2、 0 状态报警：变量的值由 1 变为 0 时产生报警；
- 3、 状态变化报警：变量的值有 0 变为 1 或由 1 变为 0 为都产生报警；

11.3 事件类型及使用方法

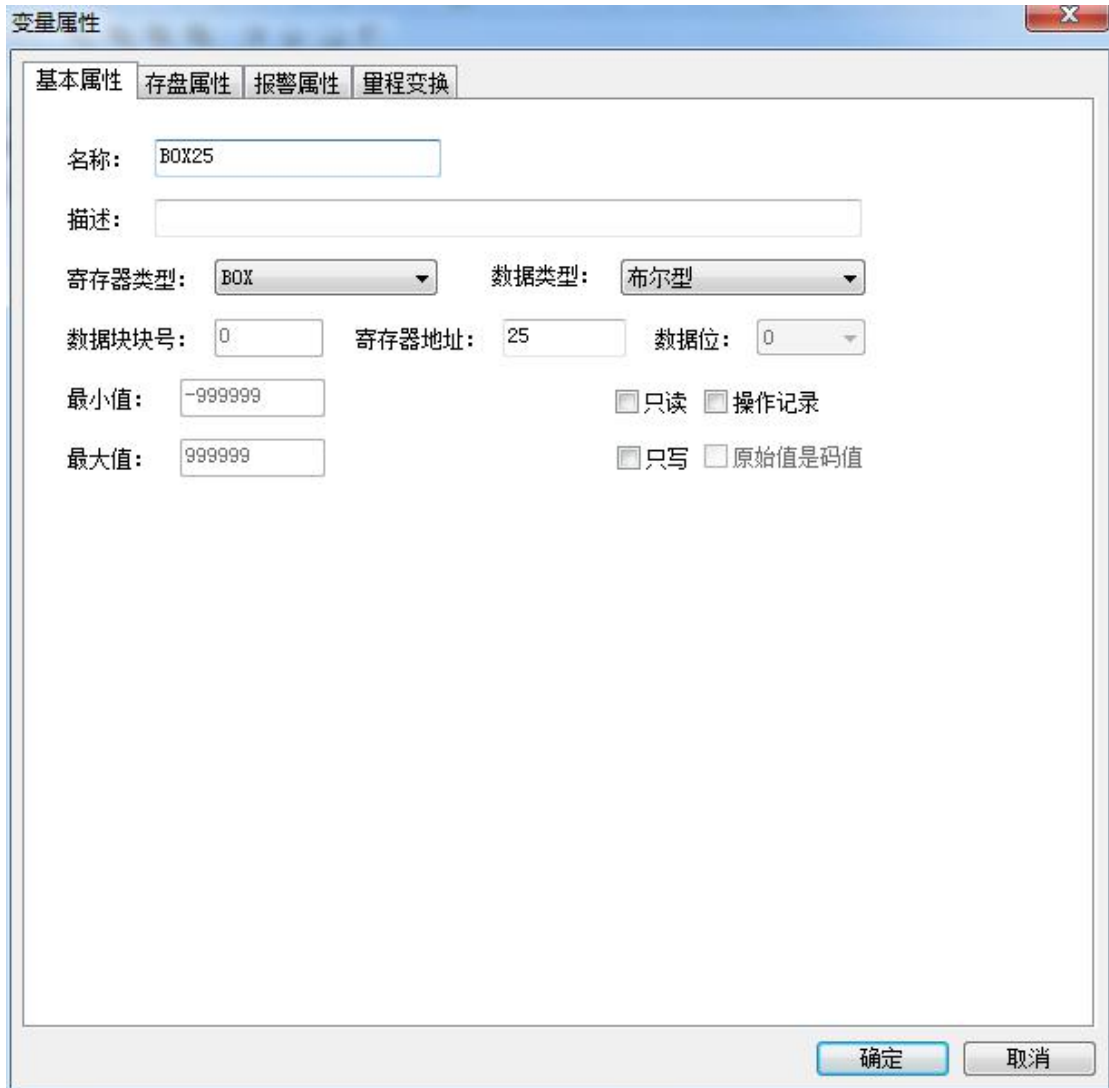
事件是不需要用户来应答的。根据操作对象和方式等的不同，事件分为以下几类：

- 1、 操作事件：用户对变量的值或变量其它域的值进行修改；
- 2、 登录事件：用户登录到系统，或从系统中退出登录；

11.3.1 操作事件

操作事件是指用户修改有“生成事件”定义的变量的值或其域的值进行修改时，系统产生的事件。如修改重要参数的值，或报警限值、变量的优先级等。操作事件可以进行记录，使用户了解当时的值是多少，修改后的值是多少。

变量要生成操作事件，必须先要定义变量的“报警”。



The image shows a software dialog box titled "变量属性" (Variable Properties). It has four tabs: "基本属性" (Basic Properties), "存盘属性" (Storage Properties), "报警属性" (Alarm Properties), and "里程变换" (Mileage Conversion). The "基本属性" tab is selected. The dialog contains the following fields and options:

- 名称 (Name): BOX25
- 描述 (Description): (empty text box)
- 寄存器类型 (Register Type): BOX
- 数据类型 (Data Type): 布尔型 (Boolean)
- 数据块块号 (Data Block Number): 0
- 寄存器地址 (Register Address): 25
- 数据位 (Data Bit): 0
- 最小值 (Minimum Value): -999999
- 最大值 (Maximum Value): 999999
- 只读 (Read-only):
- 操作记录 (Operation Record):
- 只写 (Write-only):
- 原始值是码值 (Original value is code value):

At the bottom right, there are two buttons: "确定" (OK) and "取消" (Cancel).



图 11.5 报警属性

11.3.2 用户登录事件

用户登录事件是指用户向系统登录时产生的事件。

用户登录时，如果登录成功，则产生“登录成功”事件；如果登录失败或取消登录过程，则产生“登录失败”事件；如果用户退出登录状态，则产生“注销”事件。

11.4 报警图元

运行系统中报警的实时显示是通过报警图元实现的。报警图元分为两类：实时报警图元和历史报警图元。实时报警图元主要显示当前系统中存在的符合报警图元显示配置条件的实

时报警信息和报警确认信息及系统中的事件。历史报警窗显示当前系统中符合报警窗显示配置条件的所有报警和事件信息。报警窗口中最大显示的报警条数取决于最大报警行数的设置。

11.5 语音报警

为有效的提醒现场工作人员，CORWARE 系统提供了播放语音文件的支持，目前支持 WAV 文件。为保证空间使用效率，注意将 WAV 格式转换为 PCM 8.000kHz，8 声位，单声道格式。

播放语音脚本语法格式：`void PlaySound(LPCTSTR strFileName)`。

`strFileName` ， WAV 文件名称。

例程：要在播放 `alarm.wav` 文件，`document.PlaySound "alarm.wav"`。

清除语音文件语法格式：`void CleanWavData()`。

例程：要在清除人机界面中的语音文件，`document.CleanHisData`。

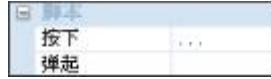
注意语音报警文件在 `audio` 目录下，不可以放在其他目录下。

第十二章 脚本语言

CORWARE 软件支持标准 VBScript 脚本，支持图元脚本、画面脚本、全局脚本。

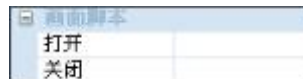
12.1 图元脚本

图元脚本分为按下、弹起两种动作。



12.2 画面脚本

画面脚本分为打开、关闭两种动作。



12.3 全局脚本

全局脚本分为定时执行、触发执行两种动作。



图 12.3 工程管理器全局脚本



图 12.3 脚本编辑器

12.3.1 定时执行

根据设定的定时周期，执行脚本，注意启动时定时执行自动执行一次。

12.3.2 触发执行

根据设定的触发条件，执行脚本，可以选择触发条件为“值从 0 变非 0 时执行”和“值从非 0 变 0 时执行”。

12.4 VBScript

12.4.1 VBScript 变量

VBScript 变量用于保存值或表达式。

变量可以有一个短的名称，如 `x`，或一个更具描述性的名称，如 `carname`。

VBScript 变量名称的规则：

- 必须以字母开头
- 不能包含点号 (.)
- 不能超过 255 个字符

在 VBScript 中，所有的变量都与类型 *variant* 相关，可存储不同类型的数据。

声明（创建）VBScript 变量

在 VBScript 创建变量通常指"声明"变量。

您可以通过 `Dim`、`Public` 或 `Private` 语句声明 VBScript 变量。如下所示：

```
Dim x  
Dim carname
```

现在您已经创建了两个变量。变量的名称是 "x" 和 "carname"。

您也可以在脚本中通过使用它的名称来声明变量。如下所示：

```
carname="Volvo"
```

现在您又创建了一个变量。变量的名称是 "carname"。然后，这个做法不是一个好习惯，因为您可能会在脚本中拼错变量名，那样可能会在脚本运行时引起奇怪的结果。

如果您拼错变量名，比如 "carname" 变量错拼为 "carnime"，脚本会自动创建一个名为 "carnime" 的新变量。为了防止脚本这样做，您可以使用 `Option Explicit` 语句。如果您使用这个语句，就必须使用 `dim`、`public` 或 `private` 语句来声明所有的变量。

把 `Option Explicit` 语句放置于脚本的顶端，如下所示：

```
Option Explicit  
Dim carname  
carname=some value
```

为变量赋值

您可以为某个变量赋值，如下所示：

```
carname="Volvo"  
x=10
```

变量名是在表达式的左侧，需要赋给变量的值在表达式的右侧。现在变量 "carname" 的值是 "Volvo"，变量 "x" 的值是 "10"。

变量的生存期

变量的生存期指的是它可以存在的时长。

当您在一个子程序中声明变量时，变量只能在此程序内进行访问。当退出此程序时，变量也会失效。这样的变量称为本地变量。您可以在不同的子程序中使用名称相同的本地变量，因为每个变量只能在声明它的程序内得到识别。

如果您在子程序以外声明了一个变量，在您的页面上的所有子程序都可以访问它。这类变量的生存期始于它们被声明，止于页面被关闭。

VBScript 数组变量

数组变量用于在一个单一的变量中存储多个值。

在下面的实例中，声明了一个包含 3 个元素的数组：

```
Dim names(2)
```

括号内显示的数字是 2。数组的下标以 0 开始，因此该数组包含 3 个元素。这是容量固定的数组。您可以为数组的每个元素分配数据，如下所示：

```
names(0)="Tove"  
names(1)="Jani"  
names(2)="Stale"
```

同样地，通过使用特定数组元素的下标号，您可以取回任何元素的值。如下所示：

```
mother=names(0)
```

您可以在一个数组中使用多达 60 个维数。声明多维数组的方法是在括号中用逗号来分隔数字。这里，我们声明了一个包含 5 行 7 列的 2 维数组：

为二维数组赋值:

```
Dim tDim x(2,2)
x(0,0)="Volvo"
x(0,1)="BMW"
x(0,2)="Ford"
x(1,0)="Apple"
x(1,1)="Orange"
x(1,2)="Banana"
x(2,0)="Coke"
x(2,1)="Pepsi"
x(2,2)="Sprite"
for i=0 to 2
document.write("<p>")
for j=0 to 2
document.write(x(i,j) & "<br />")
next
document.write("</p>")
next
```

12.4.2 条件语句

条件语句用于根据不同的情况执行不同的操作。

在 VBScript 中, 我们可以使用四种条件语句:

- **If ..Then** 语句 - 假如您希望在条件为 **true** 时执行一系列的代码, 可以使用这个语句
- **If...Then...Else** 语句 - 假如您希望执行两套代码其中之一, 可以使用这个语句
- **If...Then...ElseIf** 语句 - 假如您希望选择多套代码之一执行, 可使用这个语句
- **Select Case** 语句 - 假如您希望选择多套代码之一来执行, 可以使用这个语句

If...Then...End If

- 在条件为 **true** 时, 执行某段代码

```
If a=1 Then
document.SetPointValue "s2",CStr("234"),False
End If
```

在上面的代码中,在条件为 `true` 时(当 `a=1` 时),将“234”赋值给“s2”。

假如您想要在条件为 `true` 时执行某条语句,并在条件不为 `true` 时执行另一条语句,就必须添加关键词 `"Else"`:

If...Then...Else...End If

- 选择两段代码之一来执行

```
If a=1 Then
document.SetPointValue "s2",CStr("234"),False
Else
document.SetPointValue "s2",CStr("235"),False
End If
```

如果在条件为 `true` 时执行不止一条语句,那么可以写多行语句

```
If a=1 Then
document.SetPointValue "s2",CStr("234"),False
document.SetPointValue "s3",CStr("234"),False
Else
document.SetPointValue "s2",CStr("235"),False
End If
```

在上面的代码中,当条件为 `true` 时会执行第一段代码,当条件不成立时执行第二段代码(当 `a` 不等于 1 时)。

如果您想要选择多套代码之一来执行,可以使用 `If...Then...ElseIf` 语句:

If...Then...ElseIf...Then...Else...End If

```
i=hour(time)
If i = 10 Then
document.SetPointValue "s3",CStr("Just started...!"),False
ElseIf i = 11 Then
document.SetPointValue "s3",CStr("Hungry!"),False
```

```

ElseIf i = 12 Then
document.SetPointValue "s3",CStr("Ah, lunch-time!"),False
ElseIf i = 16 Then
document.SetPointValue "s3",CStr("Time to go home!"),False
Else
document.SetPointValue "s3",CStr("Unknown"),False
End If

```

如果您想要选择多套代码之一来执行，可以使用 "Select Case" 语句：

Select Case

```

d=weekday(date)
Select Case d
Case 1
document.SetPointValue "s3",CStr("Sleepy Sunday"),False
Case 2
document.SetPointValue "s3",CStr("Monday again!"),False
Case 3
document.SetPointValue "s3",CStr("Just Tuesday!"),False
Case 4
document.SetPointValue "s3",CStr("Wednesday!"),False
Case 5
document.SetPointValue "s3",CStr("Thursday..."),False
Case 6
document.SetPointValue "s3",CStr("Finally Friday!"),False
Case Else
document.SetPointValue "s3",CStr("Super Saturday!!!!"),False
End Select

```

以上代码的工作原理：首先，我们需要一个简单的表达式（常常是一个变量），并且这个表达式会被做一次求值运算。然后，表达式的值会与每个 **Case** 中的值作比较。如果匹配，被匹配的 **Case** 所对应的代码会被执行。

12.4.3 循环语句

循环语句用于运行相同的代码块指定的次数。Looping statements are used to run the same block of code a specified number of times.

在 VBScript 中，我们可以使用四种循环语句：

- **For...Next** 语句 - 运行一段代码指定的次数
 - **For Each...Next** 语句 - 针对集合中的每个项目或者数组中的每个元素来运行某段代码
 - **Do...Loop** 语句 - 运行循环，当条件为 **true** 或者直到条件为 **true** 时
 - **While...Wend** 语句 - 不要使用这个语句 - 请使用 **Do...Loop** 语句代替它
-

For...Next 循环

使用 **For...Next** 语句运行一段代码指定的次数。

For 语句规定计数变量 (**i**) 以及它的初始值和结束值。**Next** 语句会以 **1** 作为步进值来递增变量 (**i**)。

实例

```
For i = 0 To 5  
some code  
Next
```

Step 关键词

通过 **Step** 关键词，您可以规定计数变量递增或递减的步进值。

在下面的实例中，计数变量 (**i**) 每次循环的递增步进值为 **2**。

```
For i=2 To 10 Step 2  
some code  
Next
```

如果要递减计数变量，您就必须使用负的 **Step** 值。并且必须规定小于开始值的结束值。

在下面的实例中，计数变量 (**i**) 每次循环的递减步进值为 **2**。

```
For i=10 To 2 Step -2
some code
Next
```

退出 For...Next

您可以通过 `Exit For` 关键词退出 `For...Next` 语句。

```
For i=1 To 10
If i=5 Then Exit For
some code
Next
```

For Each...Next 循环

For Each...Next 针对集合中的每个项目或者数组中的每个元素来重复运行某段代码。

实例

```
Dim cars(2)
cars(0)="Volvo"
cars(1)="Saab"
cars(2)="BMW"

For Each x In cars
document.write(x & "<br />")
Next
```

Do...Loop

如果你不知道重复多少次，可以使用 `Do...Loop` 语句。

Do...Loop 语句重复执行某段代码直到条件是 true 或条件变成 true。

重复执行代码直到条件是 true

您可以使用 While 关键字来检查 Do... Loop 语句的条件。

```
Do While i>10  
some code  
Loop
```

如果 **i** 等于 9，上述循环内的代码将终止执行。

```
Do  
some code  
Loop While i>10
```

这个循环内的代码将被执行至少一次，即使 **i** 小于 10。

重复执行代码直到条件变成 true

您可以使用 Until 关键字来检查 Do...Loop 语句的条件。

```
Do Until i=10  
some code  
Loop
```

如果 **i** 等于 10，上述循环内的代码将终止执行。

```
Do  
some code  
Loop Until i=10
```

这个循环内的代码将被执行至少一次，即使 **i** 等于 10。

退出 Do...Loop

您可以通过 Exit Do 关键词退出 Do...Loop 语句。

```

Do Until i=10
i=i-1
If i<10 Then Exit Do
Loop

```

这个循环内的代码，只要 **i** 不为 10 且 **i** 大于 10 时都将被执行。

12.4.4 实例

1. 返回 0-99 之间的随机数

```

randomNumber=Int(100 * Rnd())
document.SetPointValue "L7",CInt(randomNumber),False

```

Rnd()返回[0,1)之间的一个随机数

2. 在一段文本中插入变量的值

```

Dim name
name="Jan Egil"
document.SetPointValue "s3",CStr("My name is: " & name),False

```

将文本 “My name is:Jan Egil ” 赋值给变量 “s3”

3. 子程序

```

Sub mySub()
msgbox("This is a Sub procedure")
End Sub
Call mySub()

```

4. 函数程序

```

Function myFunction()

```

```
myFunction = "BLUE"
End Function
document.SetPointValue "s3",CStr("My color is: " &Function()),False
```

运行时，将文本“My color is:BLUE ”赋值给变量“s3”

5.大写或小写字符

```
txt="Have a nice day!"
document.SetPointValue "s3",CStr(UCase(txt)),False
document.SetPointValue "s4",CStr(LCase(txt)),False
```

运行时，将文本“HAVE A NICE DAY!”赋值给变量“s3”，将文本“have a nice day!”赋值给“s4”。

6.显示中文星期

```
a=WeekdayName(weekday(Date))
document.SetPointValue "s3",CStr(a),False
```

运行时，将当前星期“星期*”赋值给变量“s3”

7.显示中文月份

```
a=MonthName(Month(Date))
document.SetPointValue "s3",CStr(a),False
```

运行时，将当前月份“*月”赋值给变量“s3”

8.倒记秒数到 3000 年

```
a=DateDiff("s",now,"3000-1-1 0:0:0")
document.SetPointValue "s2",CStr("距离 3000 年还有:&a&"秒"),False
```

运行时，将当前时间到到 3000 年的秒数差赋值给 s2，函数“DateDiff”

计算时间差。单位与字符对应：年“yyyy”、季度“q”、月“m”、

日“d”、时“h”、分“n”、秒“s”、一周的日数“w”、周“ww”、

一年的日数 “y”

9.向日期增加一个时间间隔

```
a=DateAdd("d",30,Date())
document.SetPointValue "s3",CStr(a),False
```

运行时，将 30 日后的日期赋值给 “s3”。

10.这是一个日期吗？

```
somedate="10/30/99"
a=IsDate(somedate)
document.SetPointValue "s3",CStr(a),False
```

运行时，赋值 “Ture” 给 “s3”。

11.删除字符串端部的空格

```
a=Trim(" ABCDEF ")
document.SetPointValue "D0",CStr(a),False
```

运行时，赋值 “ABCDEF” 给 “D0”。

12.从一段字符串的左侧或右侧返回指定数目的字符

```
sometext="Welcome to our Web Site!!"
document.SetPointValue "s3",CStr(Left(sometext,5)),False
document.SetPointValue "s4",CStr(Right(sometext,5)),False
```

运行时，赋值 “welco” 给 “s3”，赋值 “ite!!” 给 “s4”。

12.4.5 关键字

Empty 用于指示一个未初始化的变量值。当第一次创建变量时或变量值显式设置为空时，

变量值未初始化且变量为被赋值。

实例：

```
Dim x '变量 x 未初始化！
x="ff" '变量 x 不再是未初始化
x=Empty '变量 x 未初始化！
```

注意：这和 Null 不一样！！

IsEmpty 用于测试一个变量是否未初始化。

实例：`If (IsEmpty(x)) '变量 x 未初始化？`

Nothing 用于指示一个未初始化的对象值，或者把对象变量从对象分离用于释放系统资源。

实例：`Set myObject=Nothing`

Is 用于测试一个值是否是初始化的对象。

Nothing

实例：`If (myObject Is Nothing) '它是否未设置？`

注意：如果您把一个值与 Nothing 作比较，您将不会得到正确的结果！

实例：`If (myObject = Nothing) '总是错误！`

Null 用于指示变量不包含有效数据。

Null 把值设置为"无效"，Empty 则表示值"未设置"。

注意：这不同于 Empty 或 Nothing！！

实例：`x=Null 'x 不包含有效数据`

IsNull 用于测试一个值是否包含无效数据。

实例: `if (IsNull(x)) 'x 是无效的?`

True	用于指示一个布尔条件是正确的 (True 为 -1)
False	用于指示一个布尔条件是不正确的 (False 为 0)

12.5 脚本函数

12.5.1 系统函数

1、ChangePassword

注释: 该函数用来更改设置用户密码。

返回值: 无

例子: `document.ChangePassword"管理员"`

运行时, 弹出对话框, 输入旧密码、新密码、确认新密码。点击确定, 用户“管理员”密码设置完成。

2、ChangeRecipeName

注释: 该函数用来修改配方组下指定编号配方的名称。

返回值: 无

例子: `document.ChangeRecipeName"生产面包","0","L 奶油面包"`

运行时, 将生产面包配方组“生产面包”下编号为“0”的配方名

称修改为“L 奶油面包”。

配方组下配方名 默认编号如下图红色字符所示。

	变量名	配方1	配方2	配方3	配方4
配方名		A 0	B 1	C 2	D 3
变量1	P1	0	0	0	0
变量2	P2	0	0	0	0

3、CleanHisData

注释：该函数用来清理删除历史数据。

返回值：无

例子：document.CleanHisData

运行时，会清理删除人机界面“NANDFLASH\HisData”目录下 alarm 和 trend 两个文件内容。

4、CleanWaveData

注释：该函数用来删除语音文件。

返回值：无

例子：document.CleanWaveData

运行时，清理删除人机界面中语音文件。

5、ClearTrendData

注释：该函数用来置 0 XY 曲线变量。

返回值：无

例子：document.ClearTrendData "Obj148"

运行时，XY 轴曲线变量置 0。

6、CloseWindow

注释：该函数用来关闭窗口。

返回值：无

例子：document.CloseWindow("窗口 1")

关闭窗口 1。

7、CopyDirectory

注释：该函数用来复制目录。

返回值：无

例子：

```
document.CopyDirectory"NANDFLASH\1.txt","NANDFLASH\2.txt"
```

运行时，将名为“1”的文本档内的内容复制到名为“2”的文本档中。

8、CopyHisData

注释：该函数用来复制历史报警和历史数据文件到 U 盘或 SD 卡。

返回值：无

例子：document.CopyHisData 0, 0, 2016, 5

运行时，复制 2016 年 5 月的历史数据到 U 盘。

document.CopyHisData 参数 1, 参数 2, 参数 3, 参数 4

参数 1: 0 为 U 盘, 1 为 SD 卡。

参数 2: 0 为不删除源文件, 1 为删除源文件

参数 3: 历史数据年份

参数 4: 历史数据月份

注意: 该函数只能导出整月的历史数据, 运行时用系统变量 \$CopyAlarmPercent 和 \$CopyTrendPercent 显示历史报警和历史数据文件拷贝的百分比。

9、CopyHmiFile

注释: 该函数用来复制文件。

返回值: 无

例子:

```
document.CopyHmiFile"NANDFLASH\1.txt","NANDFLASH\2.txt"
```

运行时, 将名为“1”的文本档内的内容复制到名为“2”的文本档中。

10、CopyPrjData *

注释: 该函数用来复制项目数据到 U 盘或 SD 卡。

返回值: 无

例子: document.CopyPrjData(0)

运行时, 复制项目数据到 U 盘。0 为 U 盘, 1 为 SD 卡。

11、CreateHmiFile

注释：该函数用来创建文件。

返回值：无

例子：document.CreateHmiFile"NANDFLASH\1.txt"

运行时，在 NANDFLASH 文件夹下创建名称为“1”的文本文件。

12、CreateHMIProcess

注释：该函数用来启动进程。

返回值：无

例子：document.CreateHMIProcess"\Windows\pword.exe"

运行时，启动触摸屏内软件 Microsoft WordPad 进程。

13、CSVToRCP *

注释：该函数用来将指定文件夹下 CSV 格式文件转换成 RCP 格式文件并放到指定目标文件夹下。

例子：document.CSVToRCP "硬盘\","NANDFlash\BIN\RECIPE"

运行时，将 U 盘里的所有 CSV 格式文件转成 RCP 格式文件并将文件放到 NANDFlash\BIN\RECIPE\的文件夹下。

14、Delay

注释：延时函数（有 3 中参数类型：Delay 类型，时长/秒）

例子：延时 2 秒

document.Delay 1,2

document.Delay 2,2

document.Delay 3,2

按钮等属性框脚本中：Delay 1

循环脚本中：Delay 2 和 Delay 3

触发执行时：Delay 2 和 Delay 3

15、DeleteHmiFile

注释：该函数用来删除文件。

返回值：无

例子：document.DeleteHmiFile"NANDFLASH\1.txt"

运行时，删除 NANDFLASH 文件夹下名为“1”的文本文档。

16、Editusers

注释：该函数用来编辑用户管理。

返回值：无

例子：document.Editusers。

运行时，弹出对话框，在线新建用户、删除用户、修改用户名、修改

密码、修改操作权限等。

17、ExitSystem

暂未完善。

18、FileExists

注释：该函数用来查询后台文件夹下是否有文件存在。

返回值：无

例子：v=document.FileExists ("nandflash\bin\")

document.SetPointValue "D0",CBool(v),False

运行时，查询后台 nandflash 文件夹下 bin 目录下是否有文件，有返回 1，没有返回 0，将返回值赋值给布尔型变量 D0。

19、GetAlarmFileCount

注释：查询历史报警文件数量

返回值：短整型

例子：w=document.GetAlarmFileCount

document.SetPointValue "L3",CInt(w),False

运行后，“L3”显示历史报警文件数量

20、GetAlarmFileSize

注释：查询历史报警文件总大小

返回值：短整型

例子：w=document.GetAlarmFileSize

```
document.SetPointValue "L4",CInt(w),False
```

运行后，“L4”显示历史报警文件大小，单位 KB

21、GetDiskFreeSpace

注释：该函数用来获得人机界面中的储存器剩余空间。

返回值：短整形

例子：a=document.GetDiskFreeSpace(0)

```
b=document.GetDiskFreeSpace(1)
```

```
c=document.GetDiskFreeSpace(2)
```

```
document.SetPointValue "D1",CStr(a),False
```

```
document.SetPointValue "D2",CStr(b),False
```

```
document.SetPointValue "D3",CStr(c),False
```

运行时，获得人机界面的储存器剩余空间。0 为 FLASH；1 为 U 盘；2 为 SD 卡。将 FLASH、U 盘、SD 卡的剩余空间大小分别赋给变量 D1、D2、D3，改大小为字符型，因此要将 D1、D2、D3 的数据类型设置为字符型，数值单位为字节。

22、GetHisData

注释：该函数用来获得历史数据。

返回值：返回当前变量的数值，数值类型与变量相同。

例子：a=document.GetHisData("D100",2015,5,1,14,30,30)

document.SetPointValue "D0",CStr(a),False

运行时，获得变量 D100 在 2015 年 5 月 1 日 14 时 30 分 30 秒时刻的历史数据。将获得的历史数据赋给字符串变量 D0。

23、GetPointValue

注释：该函数用来读取变量当前值。

返回值：返回当前变量的数值，数值类型与变量相同。

例子：a=document.GetPointValue("D0")

读取变量 D0 的值，并把 D0 值赋给 a。

24、GetRecipeCount

注释：该函数用来返回配方组内配方的组数。

返回值：短整形

例子：a=document.GetRecipeCount("生产面包")

document.SetPointValue "D0",CStr(a),False

运行时，将返回值赋值给字符串变量 D0。

25、GetRecipeName

注释：该函数用来返回配方组内指定配方号的配方名称。指定配方为对应配方的编号。配方 1~n 的编号为 0~n-1。

返回值：字符型

例子: `document.SetPointValue "D0",document.GetRecipeName("生产面包",0),False`

例子中“生产面包”为配方组名，“0”为配方编号（下图红色标识）；运行时，返回值为‘A’，并将返回值赋值给字符串变量 D0。

	变量名	配方1	配方2	配方3	配方4
配方名		A 0	B 1	C 2	D 3
变量1	P1	0	0	0	0
变量2	P2	0	0	0	0

26、GetRecipeNo

注释：该函数用来返回配方组内当前配方的编号，配方不存在返回-1.

返回值：字符型

例子: `a=document.GetRecipeNo("生产面包","L 甜面包")`

`document.SetPointValue "D0",CStr(a),False`

运行时，返回值为 0,并将返回值赋值给字符串变量 D0。

27、GetRecipeValue

注释：该函数用来读取指定配方下变量的数值.

返回值：短整型

例子: `a=document.GetRecipeValue("L 生产面包","L 甜面包","L 糖")`

`document.SetPointValue "D0",CInt(a),False`

"L 生产面包"为配方组名，“L 甜面包”为配方名，“L 糖”为变量；

运行时，读取生产面包配方组下“L 甜面包”配方里变量 L 糖的数值，

并将读取中赋给变量 D0。

28、GetSystemFreeMemory

注释：该函数用来返回人机界面中的可用内存。

返回值：字符型

例子：a=document.GetSystemFreeMemory

```
document.SetPointValue "D1",CStr(a),False
```

运行时，将人机界面的可用内存大小赋给 D1，该大小为字符型，因此要将 D1 的数据类型设置为字符型，单位为字节。

29、GetTrendFileCount

注释：该函数用来返回历史数据数量。

返回值：短整型

例子：w=document.GettrendFileCount

```
document.SetPointValue "L3",CInt(w),False
```

运行后，“L3”显示历史文件数量

30、GetTrendFileSize

注释：查询历史文件总大小

返回值：短整型

例子：w=document.GetTrendFileSize

```
document.SetPointValue "L4",CInt(w),False
```

运行后，“L4”显示历史文件大小，单位 KB

31、HisAlarmExport

注释：该函数用来导出历史报警文件。

返回值：无

例子：document.HisAlarmExport"硬盘\123.txt"

运行时，将历史报警文件导出到“U 盘”名为“123”的文本文件中。

32、HisDataExport

注释：该函数用来导出历史数据，需配合系统函数\$FileOperation 使用。

返回值：无

例子：document.HisDataExport"硬盘\历史数据.csv",2015,7,1,11

运行时，将存放在触摸屏后台文件夹 hisdata\trend 里的历史数据文件“2015-7-1 11.dat”导出到“U 盘”名为“历史数据”的 CSV 格式的文件中。后台历史文件是以“年-月-日 时”的形式保存的。
\$FileOperation 为文件操作函数，当文件正在转换时为 1，转换完成时为 0。当此函数为 0 时可将 U 盘从触摸屏上拔下，否则会报错。

33、HistoricCurve_FirstPage

注释：该函数用来显示历史曲线查询的第一页。

返回值：无

例子：document.HistoricCurve_FirstPage("Obj2")

运行时，显示已查询历史曲线第一页，“Obj2”为历史曲线构件名称。

34、HistoricCurve_LastPage

注释：该函数用来显示历史曲线查询的最后一页。

返回值：无

例子：`document.HistoricCurve_LastPage("Obj2")`

运行时，显示已查询历史曲线最后一页，"Obj2"为历史曲线构件名称。

35、HistoricCurve_NextPage

注释：该函数用来显示曲线事件查询的下一页。

返回值：无

例子：`document.HistoricCurve_NextPage("Obj2")`

运行时，显示已查询历史曲线下一页，"Obj2"为历史曲线构件名称。

36、HistoricCurve_PrevPage

注释：该函数用来显示历史曲线查询的上一页。

返回值：无

例子：`document.HistoricCurve_PrevPage("Obj2")`

运行时，显示已查询历史曲线上一页，"Obj2"为历史曲线构件名称。

37、HistoricCurve_Query

注释：该函数用来设置历史曲线查询时间。

返回值：无

例子：

```
document.HistoricCurve_Query"Obj2",2015,7,7,15,30,30,2015,7,7,16,30,30
```

运行时，查询 2015-7-7 15:30:30 到 2015-7-7 16:30:30 之间的历史曲线，"Obj2"为历史曲线构件名称。

38、HistoricEvent_FirstPage

注释：该函数用来显示历史事件查询的第一页。

返回值：无

例子：document.HistoricEvent_FirstPage("Obj1")

运行时，显示已查询历史事件第一页，"Obj1"为历史报警构件名称。

39、HistoricEvent_LastPage

注释：该函数用来显示历史事件查询的最后一页。

返回值：无

例子：document.HistoricEvent_LastPage("Obj1")

运行时，显示已查询历史事件最后一页，"Obj1"为历史报警构件名称。

40、HistoricEvent_NextPage

注释：该函数用来显示历史事件查询的下一页。

返回值：无

例子：document.HistoricEvent_NextPage("Obj1")

运行时，显示已查询历史事件下一页，"Obj1"为历史报警构件名称。

41、HistoricEvent_PrevPage

注释：该函数用来显示历史事件查询的上一页。

返回值：无

例子：document.HistoricEvent_PrevPage("Obj1")

运行时，显示已查询历史事件上一页，"Obj1"为历史报警构件名称。

42、HistoricEvent_Query

注释：该函数用来设置历史事件查询时间。

返回值：无

例子：

document.HistoricEvent_Query"Obj1",2015,7,7,15,30,30,2015,7,7,16,30,
30

运行时，查询 2015-7-7 15:30:30 到 2015-7-7 16:30:30 之间发生的报警事件，"Obj1"为历史报警构件名称。

43、OpenGraph

注释：该函数用来切换画面。

返回值：无

例子：document.OpenGraph("画面 1")

打开画面 1。

44、OpenWindow

注释：该函数用来弹出窗口。

返回值：无

例子：document.OpenWindow"窗口 1",0,0,False

弹出窗口，在位置坐标 x=0,y=0 显示，不显示标题。**False** 为不显示标题，**True** 显示标题。

45、Play sound

注释：该函数用来播放音频文件

返回值：无

例子：document.PlaySound " *.wav "

音频文件路径：NANDFlash\bin\audio

<音量+>调用脚本：document.Volumeincr

<音量->调用脚本：document.Volumedecr

46、PointTable_GetCurPage

注释：该函数用来获得变量浏览图元中当前寄存器显示页数。

返回值：短整型

例子：a=document.PointTable_GetCurPage("Obj88")

```
document.SetPointValue "D0",CInt(a),False
```

运行时，将变量浏览图元 Obj88 中当前页码数赋值给 D0。

47、PointTable_GetMaxPage

注释：该函数用来获得变量浏览图元的最大页数。

返回值：短整型

例子：a=document.PointTable_GetMaxPage("Obj88")

```
document.SetPointValue "D0",CInt(a),True
```

运行时，将变量浏览图元 Obj88 中最大页码数赋值给 D0。

48、PointTable_SetFirstRowIndex

注释：该函数用来设置变量浏览图元的中起始序号。

返回值：无

例子：document.PointTable_SetFirstRowIndex "Obj88",3

运行时，设置变量浏览图元 Obj88 的起始序号为 3。

49、PointTable_SetPage

注释：该函数用来设置变量浏览图元的当前寄存器显示页数。

返回值：短整型

例子：`document.PointTable_SetPage "Obj88",3`

运行时，变量浏览图元 **Obj88** 中显示第 3 页的内容。

50、PointTable_SetRegister

注释：该函数用来设置变量浏览图元的当前寄存器类型。

返回值：无

例子：`document.PointTable_SetRegister "Obj88","D"`

运行时，在变量浏览图元 **Obj88** 中浏览所有 D 寄存器的值。

51、PointTable_SetValue

注释：该函数用来设置变量浏览图元中被选择变量的值。

返回值：无

例子：`document.PointTable_SetValue("Obj88")`

运行时，在变量浏览图元 **Obj88** 中可以设置各个变量的值。

52、RCPTOCSV

注释：该函数用来将指定文件夹下 RCP 格式配方文件转换成 CSV 格式文件并放到指定目标文件夹下。

返回值：无

例子：`document.RCPToCSV "NANDFlash\BIN\RECIPE\","硬盘\"`

运行时，将 NANDFlash\BIN\RECIPE\文件夹下所有 RCP 格式配方文件转成 CSV 格式文件并将文件放到 U 盘里。

53、ReadString

注释：该函数用来读字符串数据。

返回值：无

例子：`a=document.ReadString("NANDFLASH\1.txt",1)`

`document.SetPointValue "D0",CStr(a),False`

运行时，读取 NANDFLASH 文件夹下名为“1”的文本文件的第 1 行的数据，将数据赋给字符串变量 D0。

54、RebootHMI

注释：该函数用来重新启动。

返回值：无

例子：`document.RebootHMI`

运行时，人机界面重新启动。

55、RecipeAddNew

注释：该函数用来在配方组中最后一个配方后面追加一个新配方。

返回值：无

例子: `document.RecipeAddNew"生产面包","L 肉松面包"`

运行时, 将“生产面包”配方组中追加一个名为“L 肉松面包”的新配方, 配方各变量初始值为 0。

56、RecipeDelete

注释: 该函数用来删除配方组中指定配方名的配方。

返回值: 无

例子: `document.RecipeDelete"生产面包","L 肉松面包"`

运行时, 删除生产面包配方组下配方名为“L 肉松面包”的配方。

57、RecipeLoad

注释: 该函数用来加载指定配方名的配方。

返回值: 无

例子: `RecipeLoad"生产面包","L 甜面包"`

运行时, 将加载配方名为“L 甜面包”的配方。

58、RecipeModify

注释: 该函数用来修改指定配方下变量的数值。

返回值: 无

例子: `RecipeModify"生产面包","L 甜面包","L 糖",100`

`RecipeModify"生产面包","L 甜面包","L 盐",200`

`RecipeModify"生产面包","L 甜面包","L 面粉",300`

`RecipeModify"生产面包","L 甜面包","L 水",400`

`RecipeModify"生产面包","L 甜面包","L 蜂蜜",400`

运行时，设置生产面包配方组下“L 甜面包”配方内的各个变量：
L 糖修改为 100；L 盐修改为 200；L 面粉修改为 300；L 水修改为 400；
L 蜂蜜修改为 500。

59、RefreshTrend

60、Report_GetCellText

该函数用来读取报表单元格中的内容。

返回值：字符型

例子：`a=document.Report_GetCellText("Obj1",0,0)`

`document.SetPointValue "D0",CStr(a),False`

运行时，读取报表 **Obj1** 的第 1 行第 1 列单元格的内容赋给字符串变量 **D0**。

61、Report_Load

注释：该函数用来加载报表。

返回值：无

例子：`document.Report_Load "Obj1","报表 1"`

运行时，将报表 1 的内容加载到报表 **Obj1** 中。

62、Report_Print

注释：该函数用来打印报表。

返回值：无

例子：document.Report_Print("Obj1")

运行时，打印报表 Obj1 的内容。但要配合函数 Report_Load 使用，只有先加载才可以打印；并且触摸屏要连接打印机使用。

63、Report_Query

注释：该函数用来查询报表。

返回值：无

例子：document.Report_Query("Obj1")

运行时，查询具体时间的报表 Obj1 内容。要配合配合日报表、月报表、年报表使用。

64、Report_Save

注释：该函数用来保存报表。

返回值：无

例子：document.Report_Save("Obj1")

运行时，保存报表 Obj1 的内容到“U 盘”或“SD 卡中”。但要配合函数 Report_Load 使用，只有先加载才可以保存导出。

65、Report_SetCellFillColor

注释：该函数用来设置报表单元格中的填充颜色。

返回值：无

例子：document.Report_SetCellFillColor "Obj1",0,0,RGB(255,0,0)

运行时，将报表 Obj1 的第 1 行第 1 列单元格的填充颜色设置为红色。

66、Report_SetCellText

注释：该函数用来设置报表单元格中的文字。

返回值：无

例子：document.Report_SetCellText "Obj1",0,0,"abc"

运行时，向报表 Obj1 的第 1 行第 1 列单元格写入字符串“abc”。

67、Report_SetCellTextColor

注释：该函数用来设置报表单元格中的文字颜色。

返回值：无

例子：document.Report_SetCellTextColor "Obj1",0,0,RGB(255,0,0)

运行时，将报表 Obj1 的第 1 行第 1 列单元格的文字设置为红色。

68、ScreenPrint

注释：该函数用来打印当前画面。

返回值：无

例子：document.ScreenPrint

运行时，打印当前画面，需触摸屏连接打印机。

69、SetBacklightState

注释：该函数用来设置背光状态。

返回值：无

例子：document.SetBacklightState(**False**)

document.SetBacklightState(**True**)

运行时，设置背光状态打开或者关闭，**False** 为立即进入背光状态再次点击进入画面，**True** 为进入背光等待状态,与软件中工具—其他设置—背光关闭等待时间结合使用。

70、SetButtonState

注释：该函数用来设置按钮的弹起、按下状态。

返回值：无

例子：document.SetButtonState "Obj1",**True**

document.SetButtonState "Obj1",**False**

运行时，设置图元 **Obj1** 按下或弹起，**True** 为按下，**False** 为弹起。

71、SetBuzzerState

注释：该函数用来设置蜂鸣器状态。

返回值：无

例子：document.SetBuzzerState(**False**)

document.SetBuzzerState(**True**)

运行时，设置蜂鸣器状态打开或者关闭，**False** 为关闭，**True** 为打开。

72、SetDevice

注释：该函数用来打开或关闭设备驱动。

返回值：无

例子：打开：document.SetDevice "端口 1","设备 1",1

关闭：document.SetDevice "端口 1","设备 1",0

运行时，打开或关闭端口 1 下设备 1 的驱动，1 为打开，0 为关闭。

73、SetFillColor

注释：该函数用来设置图元的填充颜色。

返回值：无

例子：document.SetFillColor "Obj1",RGB(255,0,0)

运行时，将图元 Obj1 的填充颜色设为红色。

74、SetLineColor

注释：该函数用来设置图元的边框颜色。

返回值：无

例子：document.SetLineColor "Obj1",RGB(255,0,0)

运行时，将图元 Obj1 的边框颜色设为红色。

75、SetPanelSize

注释：该函数用来设置键盘大小。

返回值：无

例子：document.SetPanelSize 0,400

document.SetPanelSize 1,400

运行时，设置数字键盘或字符键盘大小为：宽度 400，高度 400；**0** 为数字键盘，**1** 位字符键盘。

76、SetPointValue

注释：该函数用来写入变量值。

返回值：无

例子：document.SetPointValue "D0",CInt(10),True

将数值 10 写入到变量 D0 中

注意：在将值写到变量中时，数值根据变量类型进行转换。如为短整型，则需要使用 CInt 函数。CBool（布尔型），CLng（长整型），CDBl（双精度），CSng（单精度），CStr（字符串）。如果变量为无符号短整型或无符号长整型，参数为 True。如果变量为短整型或长整型，参数为 False。

77、SetPosition

注释：该函数用来设置图元的显示位置和大小。

返回值：无

例子：document.SetPosition "Obj1",0,0,100,100

运行时，将图元 **Obj1** 的显示设置位置为：坐标 $x=0,y=0$ ，大小为：宽度=100，高度=100。

78、SetSystemTime

注释：该函数用来设置系统时间。

返回值：无

例子：`document.SetSystemTime`

运行时，弹出窗口设定系统时间。

79、SystemTimeAuto

注释：该函数用来自动设置系统时间。

返回值：无

例子：`document.SetSystemTimeAuto 2015,05,01,12,00,00`

运行时，设定系统时间 2015 年 5 月 1 日 12 时 0 分 0 秒。

80、SetText

注释：该函数用来设置图元的文本内容。

返回值：无

例子：`document.SetText "Obj1","停止"`

运行时，将图元 **Obj1** 的文本内容设为“停止”。

注释：该函数用来设置图元的文本颜色。

返回值：无

例子：`document.SetTextColor "Obj1",RGB(255,0,0)`

运行时，将图元 **Obj1** 的文本颜色设为红色。

81、SetTextColor

82、SetTrendMAXPointNum

注释：该函数用来设置 XY 轴曲线 X 轴最大点数

注释：该函数用来设置 XY 曲线的最大点数。

返回值：无

例子：`document.SetTrendMAXPointNum "Obj1350",64`

运行时，将 XY 轴曲线图元“**Obj1350**”的 X 轴最大点数设为 64.

83、SetTrendXMAXPointValue

注释：该函数用来设置 XY 轴曲线 X 轴最大值

返回值：无

例子：`document.SetTrendXMAXPointValue "Obj1350",63`

运行时，将 XY 轴曲线图元“**Obj1350**”的 X 轴最大值设为 63.

84、SetTrendXMINPointValue

注释：该函数用来设置 XY 轴曲线 X 轴最小值

返回值：无

例子：`document.SetTrendXMINPointValue "Obj1350",0`

运行时，将 XY 轴曲线图元“Obj1350”的 X 轴最小值设为 0.

85、SetTrendXPointValue

注释：该函数用来设置 XY 轴曲线 X 轴的值

返回值：无

例子：`document.SetTrendXPointValue "Obj1350",i,1,1,1,0,0,0,0,0`

运行时，将 XY 轴曲线图元“Obj1350”的 3 条曲线的第 i 个点的 X 轴都设为 1

86、SetTrendYMAXPointValue

注释：该函数用来设置 XY 轴曲线 Y 轴最大值

返回值：无

例子：`document.SetTrendYMAXPointValue "Obj1350",100`

运行时，将 XY 轴曲线图元“Obj1350”的 Y 轴最大值设为 100.

87、SetTrendYMINPointValue

注释：该函数用来设置 XY 轴曲线 Y 轴最小值

返回值：无

例子：`document.SetTrendYMINPointValue "Obj1350",0`

运行时，将 XY 轴曲线图元“Obj1350”的 Y 轴最小值设为 0.

88、SetTrendYPointValue

注释：该函数用来设置 XY 轴曲线 Y 轴的值

返回值：无

例子：document.SetTrendYPointValue "Obj1350",i,y1,y2,y3,0,0,0,0,0

运行时，将 XY 轴曲线图元“Obj1350”的 3 条曲线的第 i 个点的 Y 轴分别设为 y1，y2，y3.

89、SetVisible

注释：该函数用来设置图元的可见性。

返回值：无

例子：document.SetVisible "Obj1",True

document.SetVisible "Obj1",False

运行时，设置图元 Obj1 可见或不可见，True 为可见，False 为不可见。

90、UpdateMHSystem *

注释：该函数用来更新程序到设备。

返回值：无

例子：document.UpdateMHSystem

运行时，选择 U 盘或者 SD 卡更新。

91、UserLogin

注释：该函数用来登录用户。

返回值：无

例子: `document.UserLogin`

如果没有登录, 则函数执行时弹出登录窗口, 如果已经登录, 则退出登录。

注意: 此函数不可以在全局脚本中调用, 如果在全局脚本中调用则会出现错误, 导致程序异常退出。

92、UserLoginAuto

注释: 该函数用来自动登录用户。

返回值: 无

例子: `document.UserLoginAuto("管理员")`

运行时, 自动登陆管理员, 无需输入密码。

93、WriteString

注释: 该函数用来写字符串数据。

返回值: 无

例子: `document.WriteString"NANDFLASH\1.txt","你好",1`

运行时, 向 `NANDFLASH` 文件夹下名称为“1”的文本文件的第 1 行写入字符串“你好”。

94、SetBuzzerON

注释: 轰鸣器长鸣开关。

例子: `SetBuzzerON(1)`

`SetBuzzerON(0)`

参数等于 **1** 代表开启长鸣 参数等于 **0** 代表关闭长鸣

95、 CatchScreen

注释: 该函数用来捕捉当前画面图片

返回值: 无

例子: `document.CatchScreen 800,480`

语法格式: `void CatchScreen(int width,int high);`

width: 捕捉像素宽度, **high**: 捕捉像素高度, 默认保存路径为
`NANDFlash \ CatchScreen \ 年月日时分秒.bmp`

96、 DeleteFolder

注释: 删除文件夹下指定文件, 或者全部文件及文件夹。

返回值: 布尔型

例子: `document.DeleteFolder"NANDFlash\test*.*"`

//删除 test 下全部文件及文件夹, test 保留空文件夹

例子: `document.DeleteFolder"NANDFlash\test\alarm"`

//删除 alarm 文件夹及文件夹下所有文件

97、 VideoPlayer

注释: 用专用视频播放器播放视频文件。

返回值: 无

例子: `document.VideoPlayer "NANDFlash\Videotest.wmv"`

运行时, 播放 NANDFlash\Videotest.wmv 视频文件;

视频文件需为 .wmv 格式, 同时需硬件版本支持。专用视频播放器名称和存放的位置: \NANDFlash\TCPMP\PLAYER.exe

98、 ImageViewer

注释: 浏览固定目录下的 jpg 和 bmp 文件 (可前后翻页预览)。

返回值: 无

例子: `document.ImageViewer`

图片固定目录为: NANDFlash\Photo\

专用视频播放器名称和存放的位置:

NANDFlash\ImageViewer\ImageView.exe

99、 GetFileFolderCount

注释: 查询得到所查询路径下的总文件的数量。

返回值: 短整形

例子: `e=document.GetFileFolderCount("NANDFlash\photo")`

`document.SetPointValue "FolderCount",CInt(e),False`

FolderCount 为中间变量, 可将这个变量数值显示出来

100、GetFileFolderSize

注释：查询得到所查询路径下的总文件的大小。

返回值：短整形

例子：f=document. GetFileFolderSize("NANDFlash\photo")

```
document.SetPointValue "FolderSize",CInt(f),False
```

FolderSize 为中间变量，可将这个变量数值显示出来

101、ChangeMasterComSetup

注释：根据串口号修改其他参数，不能修改串口号本身；修改完后，需要软重启 HMI（document.RebootHMI）。

返回值：无

语法格式：

```
void ChangeMasterComSetup(int m_sPort,long m_nBaud,int m_nDataBits, int m_nParity, int m_nStopBits,int m_nTimeouts)
```

参数：m_sPort：端口号（1-8）；

m_nBaud：波特率（4800,9600,19200,38400,57600，115200）

m_nDataBits：数据位（8,7）

m_nParity：校验位（0,1,2）0 表示无校验，1 表示奇校验，2 表示偶校验

m_nStopBits：停止位（1,2）

m_nTimeouts: 超时时间（默认 500，可修改）

例子: m_sPort=document.GetPointValue("m_sPort");//端口号

m_nBaud=document.GetPointValue("m_nBaud");//波特率

m_nDataBits=document.GetPointValue("m_nDataBits");//数据位

m_nParity=document.GetPointValue("m_nParity");//校验位

m_nStopBits=document.GetPointValue("m_nStopBits");//停止位

m_nTimeouts=document.GetPointValue("m_nTimeouts");//超时时

间

document.ChangeMasterComSetupm_sPort,m_nBaud,m_nDataBits,m_n
Parity,

m_nStopBits,m_nTimeouts

102、ChangeSlaveComSetup

注释：修改 modbus 从站端口参数，修改完后，需要软重启 HMI
(document.RebootHMI)。

返回值：无

语法格式：

```
void ChangeSlaveComSetup(int m_sPort, long m_nBaud, int
    m_nDataBits, int m_nParity, int m_nStopBits, int m_nAdd
    r)
```

参数: m_sPort: 端口号 (1-8) ;

m_nBaud: 波特率(4800, 9600, 19200, 38400, 57600, 115200)

m_nDataBits: 数据位 (8, 7)

m_nParity: 校验位 (0, 1, 2) 0 表示无校验, 1 表示奇校验,
2 表示偶校验

m_nStopBits: 停止位 (1, 2)

m_nAddr: 从站站号 (0-255)

例子: m_sPort=document.GetPointValue("m_sPort")//端口号

m_nBaud=document.GetPointValue("m_nBaud")//波特率

m_nDataBits=document.GetPointValue("m_nDataBits")//数
据位

m_nParity=document.GetPointValue("m_nParity")//校验位

m_nStopBits=document.GetPointValue("m_nStopBits")//停
止位

m_nAddr=document.GetPointValue("m_nAddr") //从站站号

document.ChangeSlaveComSetupm_sPort, m_nBaud, m_nDataBits,
m_nParity, m_nStopBits, m_nAddr

103、HistoricReport_FirstPage

注释: 该函数用来显示历史表格查询的第一页.

返回值: 无

例子: `document.HistoricReport_FirstPage("Obj2")`

运行时, 显示已查询历史表格第一页, "Obj2"为历史表格构件名称。

104、HistoricReport_LastPage

注释: 该函数用来显示历史表格查询的最后一页.

返回值: 无

例子: `document.HistoricReport_LastPage("Obj2")`

运行时, 显示已查询历史表格最后一页, "Obj2"为历史表格构件名称。

105、HistoricReport_NextPage

注释: 该函数用来显示表格事件查询的下一页.

返回值: 无

例子: `document.HistoricReport_NextPage("Obj2")`

运行时, 显示已查询历史表格下一页, "Obj2"为历史表格构件名称。

106、HistoricReport_PrevPage

注释: 该函数用来显示历史表格查询的上一页.

返回值: 无

例子: `document.HistoricReport_PrevPage("Obj2")`

运行时, 显示已查询历史表格上一页, "Obj2"为历史表格构件名称。

107、HistoricReport_Query

注释：该函数用来设置历史表格查询时间.

返回值：无

例子：

```
document.HistoricReport_Query"Obj2",2015,7,7,15,30,30,2015,7,7,16,30,30
```

运行时，查询 2015-7-7 15:30:30 到 2015-7-7 16:30:30 之间的历史数据，"Obj2"为历史表格构件名称

108、HistoricReport_ExportCSV

注释：该函数用来导出历史表格已查询的数据，CSV 格式.

返回值：无

例子：

```
document.HistoricReport_ExportCSV("Obj2")
```

运行时，导出历史表格已查询的数据，按照 csv 格式导出到 u 盘，"Obj2"为历史表格构件名称

109、HistoricReport_ExportDAT

注释：该函数用来导出历史表格已查询的数据，DAT 格式.

返回值：无

例子：

```
document.HistoricReport_ExportDAT("Obj2")
```

运行时，导出历史表格已查询的数据，按照 dat 格式导出到 u 盘，
"Obj2"为历史表格构件名称

110、SetSaveType

注释：该函数用来设置存盘类型，CSV 格式。

返回值：无

例子：

`document.SetSaveType` 变量名,存盘类型,保存时间

变量名：需要存盘的变量，如“W4X01”。

存盘类型：0-不存盘，1-定时存盘，2-数据变化存盘，3-触发条件满足时存盘。

保存时间：当存盘类型为 1（定时存盘）时，才需要设置保存时间，其他的方式时存盘设置为 0。

运行时，需要把变量存盘属性中勾选定时存盘属性。

111、UserLogoff

注释：该函数用来注销用户。

返回值：无

例子：`document.UserLogoff()`

112、RecipeModifyN

注释：该函数用来配方批量修改。

返回值：无

语法格式：`document.RecipeModifyN`“配方组名”，“配方名”，“变量名前缀”，变量起始后缀，变量个数

例子: document.RecipeModifyN"模具型号", "bb", "PF", 1, 100

运行时, 把 PF1 到 PF100 共 100 个变量数值批量的写入到配方组名为“模具型号”, 配方名为“bb”的配方中。

113、DataMove

注释: 该函数用来数据转移。

返回值: 无

语法格式: document.DataMove"目标变量名", "源变量名", 变量个数

例子: For i=1 To 5

document.DataMove"B"&CStr(i), "A"&CStr(i), 5

Next

运行时, 把 B1 到 B5 的 5 个变量数值分别赋值给 A1 到 A5 的 5 个变量数据。

114、USB_CSVMToRCP

注释: U 盘导入单个配方。

语法格式: document.USB_CSVMToRCP"配方组名", "配方名"。

例子: document.USB_CSVMToRCP "包芯

", document.GetPointValue("\$CurrentRecipeName")

document.RecipeLoad"包芯纱

", document.GetPointValue("\$CurrentRecipeName")

115、RCPToUSB_CSV

注释: 导入单个配方到 U 盘。

语法格式: document.RCPToUSB_CSV"配方组名", "配方名"。

例子: `a=document.GetPointValue("$CurrentRecipeName")`

`b=document.GetPointValue("11")`

`document.RCPToUSB_CSV"数据", a, b`

其中, 变量"11"是要导出去的文件名称设置, 导出去的文件是 CSV 格式。

116、SendMessageToHMIProcess

注释: 发送消息。

语法格式: `document.SendMessageToHMIProcess"窗口名", MGS, WP, LP`

例子: `document.SendMessageToHMIProcess"PLCDrv-1", 2, 0, 0`

表示发送一个关闭 "PLCDrv-1.exe" 执行文件的消息。

12.5.2 时间日期函数

1、Date 返回当前的系统日期；

Now 返回当前的系统日期和时间；

Year 返回一个代表年份的数字；

Month 返回一个代表月份的数字；

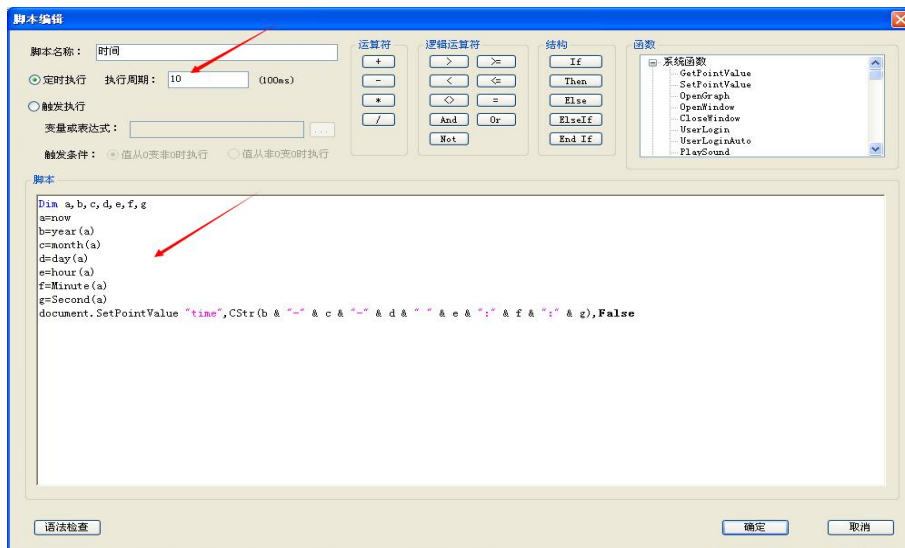
Day 返回一个代表天数的数字；

Hour 返回一个代表小时的数字；

Minute 返回一个代表月份的数字；

Second 返回一个代表秒的数字。

例子：



通过新建一个全局脚本，执行周期 10（单位 100ms），实现日期和时间的在线显示。脚本每 1S 执行一次，如下：

a=now

b=year(a)

c=month(a)

```
d=day(a)
```

```
e=hour(a)
```

```
f=Minute(a)
```

```
g=Second(a)
```

```
document.SetPointValue "time",CStr(b & "-" & c & "-" & d & " " & e  
& ":" & f & ":" & g),False
```

运行时，将日期和时间赋给字符串变量 `time`，将 `time` 通过文本输出实时显示。

2、DateAdd

注释：该函数用来添加指定时间间隔的日期。

返回值：字符型

例子：a=DateAdd("m",1,"2015-11-11 11:11:11")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，将括号中的日期时间 2015 年 11 月 11 日 11 时 11 分 11 秒加上 1 个月，并将加完后的日期时间赋给字符串变量 `D0`。其中年“yyyy”、季度“q”、月“m”、日“d”、时“h”、分“n”、秒“s”、一周的日数“w”、周“ww”、一年的日数“y”。

3、DateDiff

注释：该函数用来返回两个日期之间的时间间隔数。

返回值：字符型

例子: `a=DateDiff("yyyy","2014-11-11 11:11:11","2016-11-11 11:11:11")`

`document.SetPointValue "D0",CStr(a),False`

运行时, 比较 2 个日期的间隔年数, 并将返回值赋给字符串变量 `D0`。其中年 “`yyyy`”、季度 “`q`”、月 “`m`”、日 “`d`”、时 “`h`”、分 “`n`”、秒 “`s`”、一周的日数 “`w`”、周 “`ww`”、一年的日数 “`y`”。

4、DatePart

注释: 该函数用来返回给定日期的指定部分。

返回值: 字符型

例子: `a=DatePart("yyyy","2016-11-11 11:11:11")`

`document.SetPointValue "D0",CStr(a),False`

运行时, 返回括号中日期的年数, 并将返回值赋给字符串变量 `D0`。其中年 “`yyyy`”、季度 “`q`”、月 “`m`”、日 “`d`”、时 “`h`”、分 “`n`”、秒 “`s`”、一周的日数 “`w`”、周 “`ww`”、一年的日数 “`y`”。

5、DateSerial

注释: 该函数用来返回日期的指定年、月、日。

返回值: 字符型

例子: `a=DateSerial(2020-5,6-1,5-4)`

`document.SetPointValue "D0",CStr(a),False`

运行时, 返回日期 2015 年 5 月 1 日, 并将返回值赋给字符串变量 `D0`。

6、DateValue

注释：该函数用来返回日期。

返回值：字符型

例子：a=DateValue("September 11, 2015")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回日期 2015 年 9 月 11 日，并将返回值赋给字符串变量 D0。

7、IsDate

注释：该函数用来返回可指示计算表达式能否转换为日期的布尔型。

返回值：布尔型

例子：a=IsDate("2015,5,1")

```
b=IsDate("Hello")
```

```
document.SetPointValue "D0",CStr(a),False
```

```
document.SetPointValue "D1",CStr(b),False
```

运行时，判断字符串"2015,5,1"和 "Hello"是否为日期，并将返回值分别赋给字符串变量 D0 和 D1。D0 返回 "True",D1 返回 "False"。

8、MonthName

注释：该函数用来返回指定月份的名称。

返回值：字符型

例子: `a=MonthName(10,True)`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值“十月”, 并将返回值赋给字符串变量 `D0`。

9、Time

注释: 该函数用来返回当前的系统时间。

返回值: 字符型

例子: `a=Time`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回当前时间, 并将返回值赋给字符串变量 `D0`。

10、Timer

注释: 该函数用来返回 12:00 AM 以来的秒数。

返回值: 字符型

例子: `a=Timer`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回秒数值, 并将返回值赋给字符串变量 `D0`。

11、TimeSerial

注释: 该函数用来返回特定的小时、分钟和秒的时间。

返回值: 字符型

例子: `a=TimeSerial(20-8,30-18,30-18)`

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回日期时间 12 时 12 分 12 秒，并将返回值赋给字符串变量 **D0**。

12、TimeValue

注释：该函数用来返回时间。

返回值：字符型

例子：a=TimeValue("3:30:30 PM")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回时间 15 时 30 分 30 秒，并将返回值赋给字符串变量 **D0**。

13、Weekday

注释：该函数用来返回一个数值，代表星期的某一天。

返回值：字符型

例子：a=Weekday(2015-05-01)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回数值 7，并将返回值赋给字符串变量 **D0**。

14、WeekdayName

注释：该函数用来返回一个数值，代表星期的某一天。

返回值：字符型

例子：a=WeekdayName(6,True)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回“星期五”，并将返回值赋给字符串变量 D0。

12.5.3 类型转换函数

1、Asc

注释：该函数把字符串中首字母转换为 ANSI 字符代码。

返回值：字符型

例子：a=Asc("A")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 65，并将返回值赋给字符串变量 D0。

2、CBool

注释：该函数把表达式转换为布尔类型。

返回值：布尔型

例子：a=CBool(A=B)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 **False**，并将返回值赋给字符串变量 **D0**。

3、CByte

注释：该函数把表达式转换为字节类型。

返回值：整型

例子：a=CByte(44.77)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 45，并将返回值赋给字符串变量 **D0**。

4、CCur

注释：该函数把表达式转换为货币类型。

返回值：字符串型

例子：a=CCur(44.77123)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 44.7712，并将返回值赋给字符串变量 **D0**。

5、CDate

注释：该函数把有效的日期和时间表达式转换为日期类型。

返回值：字符串型

例子：a=CDate("October 10, 2015 10:10:10 PM")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 2015-10-10 22:10:10，并将返回值赋给字符串变

量 **D0**。

6、Cdbl

注释：该函数把表达式转换为双精度类型。

返回值：双精度浮点型

例子：a=Cdbl(444.7123*0.01)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 4.447123，并将返回值赋给字符串变量 **D0**。

7、Chr

注释：该函数把指定的 ANSI 字符代码转换为字符。

返回值：字符型

例子：a=Chr(65)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 A，并将返回值赋给字符串变量 **D0**。

8、CInt

注释：该函数把表达式转换整数类型。

返回值：整型

例子：a=CInt(44.7123)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 45，并将返回值赋给字符串变量 **D0**。

9、CLng

注释：该函数把表达式转换长整数类型。

返回值：长整型

例子：a=CLng(444444.7123)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 444445，并将返回值赋给字符串变量 D0。

10、CSng

注释：该函数把表达式转换单精度类型。

返回值：单精度浮点型

例子：a=CSng(44.71234567)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 44.71235，并将返回值赋给字符串变量 D0。

11、CStr

注释：该函数转化一个表达式为字符串

返回值：字符串型

例子：a=CStr(123)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值字符串“123”，并将返回值赋给字符串变量 D0。

12、Hex

注释：该函数返回指定数字的十六进制值。

返回值：字符串型

例子：a=Hex(10)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 A，并将返回值赋给字符串变量 D0。

13、Oct

注释：该函数返回指定数字的八进制值。

返回值：Variant (String)型

例子：a=Oct(8)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 10，并将返回值赋给字符串变量 D0。

12.5.4 格式化函数

1、FormatCurrency

注释：该函数将表达式格式化为货币值，使用系统控制面板中定义的货币符号。

返回值：字符串型

例子：a=FormatCurrency(50.5446, 3)

```
document.SetPointValue "D1",CStr(a),False
```

运行时，返回值 ¥50.545，并将返回值赋给字符串变量 D1；3 表示保留 3 位小数

2、FormatDateTime

注释：函数格式化并返回一个有效的日期或时间的表达式。

返回值：字符串型

例子：a=FormatDateTime("October 10, 2015 10:10:10 PM")

```
document.SetPointValue "s2",CStr(a),False
```

运行时，返回 2015-10-10 22:10:10,并将返回值赋给字符串变量 s2

3、FormatNumber

注释：函数返回作为数字进行格式化的表达式

返回值：字符串型

例 1：a=FormatNumber(2000)

```
document.SetPointValue "s2",CStr(a),False
```

运行时，返回 2,000.00,并将返回值赋给字符串变量 s2

例 2：a=FormatNumber(2000,3)

```
document.SetPointValue "s2",CStr(a),False
```

运行时，返回 2,000.000,并将返回值赋给字符串变量 s2.

参数 3 表示保留的小数位。

例 3：a=FormatNumber(.543,3,0)

```
b=FormatNumber(.543,3,-1)
```

运行时, a=.543,b=0.543, 参数 0 表示小数点前不加 0, 参数-1 表示小数点前加 0。

例 4: a=FormatNumber(-50,,0)

```
b=FormatNumber(-50,,,-1)
```

运行时, a=-50.00,b=(50.00), 参数 0 表示负号不放在括号中, 参数-1 表示负号放在括号中。

例 5: a=FormatNumber(1000000,,,,0)

```
b=FormatNumber(1000000,,,,,-1)
```

运行时, a=1000000.00,b=1,000,000.00, 参数 0 表示不将数字分组, 参数-1 表示将数字分组。

4、FormatPercent

注释: 函数返回作为百分数被格式化的表达式

返回值: 字符串型

例 1: a=FormatPercent(6/345)

```
document.SetPointValue "s2",CStr(a),False
```

运行时, 返回 1.74%,并将返回值赋给字符串变量 s2

例 2: a=FormatPercent(6/345,1)

```
document.SetPointValue "s2",CStr(a),False
```

运行时, 返回 1.7%,并将返回值赋给字符串变量 s2,参数 1 表示保留一位小数。

12.5.5 数学函数

1、Abs

注释：该函数返回数字的绝对值。

返回值：整型

例子：a=Abs(-44)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 44，并将返回值赋给字符串变量 D0。

2、Atn

注释：该函数返回数字的反正切值。

返回值：双精度浮点型

例子: $a=4*\text{Atn}(1)$

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 $\pi \approx 3.1415926$, 并将返回值赋给字符串变量 **D0**。

3、Cos

注释: 该函数返回数字的余弦值。

返回值: 双精度浮点型

例子: $a=\text{Cos}(0)$

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 1, 并将返回值赋给字符串变量 **D0**。

4、Exp

注释: 该函数返回 e (自然对数的底) 的幂次方。

返回值: 双精度浮点型

例子: $a=\text{Exp}(1)$

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 2.718281828, 并将返回值赋给字符串变量 **D0**。

5、Int

注释: 该函数返回指定数字的整数部分。

返回值: 整型

例子: $a=\text{Int}(44.7128)$

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 44，并将返回值赋给字符串变量 **D0**。

6、Fix

注释：该函数返回指定数字的整数部分。

返回值：整型

例子：a=Fix(44.7128)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 44，并将返回值赋给字符串变量 **D0**。

7、Log

注释：该函数返回指定数字的自然对数。

返回值：双精度浮点型

例子：a=Log(1)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 0，并将返回值赋给字符串变量 **D0**。

8、Rnd

注释：该函数返回小于 1 但大于等于 0 的一个随机数。

返回值：双精度浮点型

例子：a=Rnd()

```
document.SetPointValue "D0",CStr(a),False
```


运行时，返回值 0.5924582，并将返回值赋给字符串变量 D0。

9、Sgn

注释：符号函数，返回参数的正负（以 1，-1，0 表示）。

返回值：整型

例子：a=Sgn(11)

b=Sgn(-11)

c=Sgn(0)

```
document.SetPointValue "D0",CStr(a),False
```

```
document.SetPointValue "D1",CStr(b),False
```

```
document.SetPointValue "D2",CStr(c),False
```

运行时，返回值为 1、-1、0，并将返回值分别赋给字符串变量 D0、D1、D2。

10、Sin

注释：该函数返回指定数字（角度）的正弦值。

返回值：双精度浮点型

例子：a=Sin(1.732)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 0.9870，并将返回值赋给字符串变量 D0。

11、Sqr

注释：该函数返回指定数字的平方根。

返回值：双精度浮点型

例子：a=Sqr(4)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 2，并将返回值赋给字符串变量 D0。

12、Tan

注释：该函数返回指定数字(角度)的正切。

返回值：双精度浮点型

例子：a=Tan(1)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 1.5574，并将返回值赋给字符串变量 D0。

12.5.6 字符串函数

1、Instr

注释：该函数返回字符串在另一字符串中首次出现的位置。检索从字符串的第一个字符开始，没有检索到则返回 0。

返回值：整型

例子: `a=InStr("abcd1234abcd5678","d")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 `4`, 并将返回值赋给字符串变量 `D0`。

2、InstrRev

注释: 该函数返回字符串在另一字符串中首次出现的位置。检索从字符串的最末字符开始。

返回值: 整型

例子: `a=InStrRev("abcd1234abcd5678","d")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 `12`, 并将返回值赋给字符串变量 `D0`。

3、LCase

注释: 该函数用于把指定字符串转换为小写。

返回值: 字符串型

例子: `a=LCase("ABCDEF")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 `abcdef`, 并将返回值赋给字符串变量 `D0`。

4、Left

注释: 该函数用于从字符串的左侧返回指定字符串的字符。

返回值: 字符串型

例子: `a=Left("ABCDEF",3)`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 `ABC`, 并将返回值赋给字符串变量 `D0`。

5、Len

注释: 该函数返回字符串中的字符个数。

返回值: 整型

例子: `a=Len("ABCDEF")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 `6`, 并将返回值赋给字符串变量 `D0`。

6、LTrim

注释: 该函数用于删除字符串左侧的空格。

返回值: 字符串型

例子: `a=LTrim(" ABCDEF")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 `ABCDEF`, 并将返回值赋给字符串变量 `D0`。

7、RTrim

注释: 该函数用于删除字符串右侧的空格。

返回值: 字符串型

例子: `a=RTrim("ABCDEF ")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 ABCDEF, 并将返回值赋给字符串变量 D0。

8、Trim

注释: 该函数用于删除字符串左侧和右侧的空格。

返回值: 字符串型

例子: `a=Trim(" ABCDEF ")`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 ABCDEF, 并将返回值赋给字符串变量 D0。

9、Mid

注释: 该函数用于从字符串返回指定数目的字符。

返回值: 字符串型

例子: `a=Mid("ABCDEF",2,1)`

```
document.SetPointValue "D0",CStr(a),False
```

运行时, 返回值 B, 从字符串第二个字符开始返回一个字符, 并将返回值赋给字符串变量 D0。

10、Replace

注释：该函数用于使用另外一个字符串替换字符串的指定部分指定的字符。

返回值：字符串型

例子：a=Replace("ABCDEY","Y","F")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 ABCDEF，并将返回值赋给字符串变量 D0。

11、Right

注释：该函数用于返回从字符串右侧开始指定数目的字符。

返回值：字符串型

例子：a=Right("ABCDEF",3)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 DEF，并将返回值赋给字符串变量 D0。

12、Space

注释：该函数用于返回指定数目的空格组成的字符串。

返回值：字符串型

例子：a="ABC"&Space(3)&"DEF"

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 ABC DEF，并将返回值赋给字符串变量 D0。

13、StrComp

注释：比较两个字符串，返回代表比较结果的值。

返回值：Variant (Integer)

例子：a=StrComp("ABC","ABCD")

b=StrComp("ABCD","ABCD")

document.SetPointValue "D0",CStr(a),False

document.SetPointValue "D1",CStr(b),False

运行时，返回值-1、0，并将返回值分别赋给字符串变量 D0、D1。

两个字符串不相等返回-1，相等返回 0，字符含中文返回 1。

14、String

注释：该函数用于返回包含指定长度的重复字符的字符串。

返回值：字符串型

例子：a=String(6,"A")

document.SetPointValue "D0",CStr(a),False

运行时，返回值 AAAAAA，并将返回值赋给字符串变量 D0。

15、StrReverse

注释：该函数用于反转字符串。

返回值：字符串型

例子：a=StrReverse("ABCDEF")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 FEDCBA，并将返回值赋给字符串变量 D0。

16、UCase

注释：该函数用于把指定的字符转换为大写。

返回值：字符串型

例子：a=UCase("abcdef")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 ABCDEF，并将返回值赋给字符串变量 D0。

12.5.6 其他函数

1、InputBox

注释：可显示对话框，用户可在其中输入文本，点击按钮输出结果。

返回值：字符串型

例子：a=InputBox("输入名字")

```
document.SetPointValue "D0",CStr(a),False
```

运行时，弹出输入框，标题为“输入名字”，在输入框中输入字符，并将输入结果赋给字符串变量 D0。

2、IsEmpty

注释：该函数用于返回一个布尔值，指示指定的变量是否已被初始化。

返回值：布尔型

例子：a=document.GetPointValue("D0")

a=**Empty**

b=IsEmpty(a)

document.SetPointValue "D1",CStr(b),**False**

运行时，D0 已被初始化,返回值 True，并将返回值赋给字符串变量 D1。

3、IsNull

注释：该函数用于返回一个布尔值，指示指定的变量是否包含无效数据。

返回值：布尔型

例子：a=document.GetPointValue("D0")

a=**Null**

b=IsNull(a)

```
document.SetPointValue "D1",CStr(b),False
```

运行时，D0 中已包含无效数据,返回值 True，并将返回值赋给字符串变量 D1。

4、IsNumeric

注释：该函数用于返回一个布尔值，指示指定的表达式是否可作为数字来计算。

返回值：布尔型

例子：a=IsNumeric("123abc")

```
b=IsNumeric("123")
```

```
document.SetPointValue "D0",CStr(a),False
```

```
document.SetPointValue "D1",CStr(b),False
```

运行时，返回值分别为 False、True，并将返回值分别赋给字符串变量 D0、D1。

5、MsgBox

注释：该函数用于显示消息框，等待用户点击按钮，并返回指定用户点击了哪个按钮的值。

```
MsgBox (prompt [,Buttons] [,Title] [,Helpfile,Context])
```

Prompt:字符串表达式，用于显示对话框中的消息。

Buttons:按钮的数目及其形式。0 只显示“确定”；1 显示“确定”“取消”按钮；2 显示“终止”“重试”“忽略”；3 显示“是”“否”“取消”；4 显示“是”“否”；5 显示“重试”“取消”等。

Title: 对话框标题栏中显示的内容。

Msgbox 返回值: 1 点击了“确定”按钮；2 点击了“取消”按钮；3 点击了“终止”按钮；4 点击了“重试”按钮；5 点击了“重试”按钮；6 点击了“是”按钮；7 点击了“否”按钮。

例子: `a="123456"`

```
b=MsgBox(a,1,"msgbox")
```

```
document.SetPointValue "D0",CStr(b),False
```



运行时，当点击按钮确定时返回值为 1，点击取消时返回值为 2，并将返回值赋给字符串变量 D0。

6、RGB

注释: 该函数用于返回一个 RGB 颜色值的数字。

返回值: 字符串型

例子: `a=RGB(255,0,0)`

```
document.SetLineColor "Obj1",a
```

运行时，将图元线条 Obj1 颜色设置为红色。

7、Round

注释：该函数用于对数进行四舍五入。

返回值：字符串型

例子：a=Round(24.55,1)

```
document.SetPointValue "D0",CStr(a),False
```

运行时，返回值 24.6，并将返回值赋给字符串变量 D0。1 表示保留一位小数。

第十三章 其他常用功能

13.1 工程下载方式

13.1.1 方法一 {以太网/IP}

13.1.1.1 硬件连接

连接方式：网线连接



13.1.1.2 启动 TPC

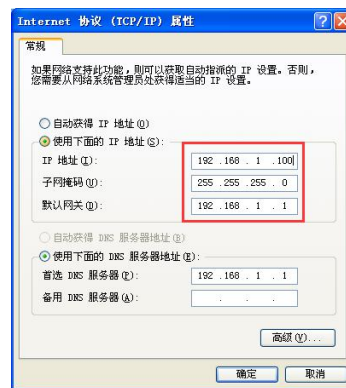
使用 24V 直流电源给 TPC 供电，开机启动后不需要任何操作，系统将自动进入工程运行界面。

13.1.1.3 工程下载

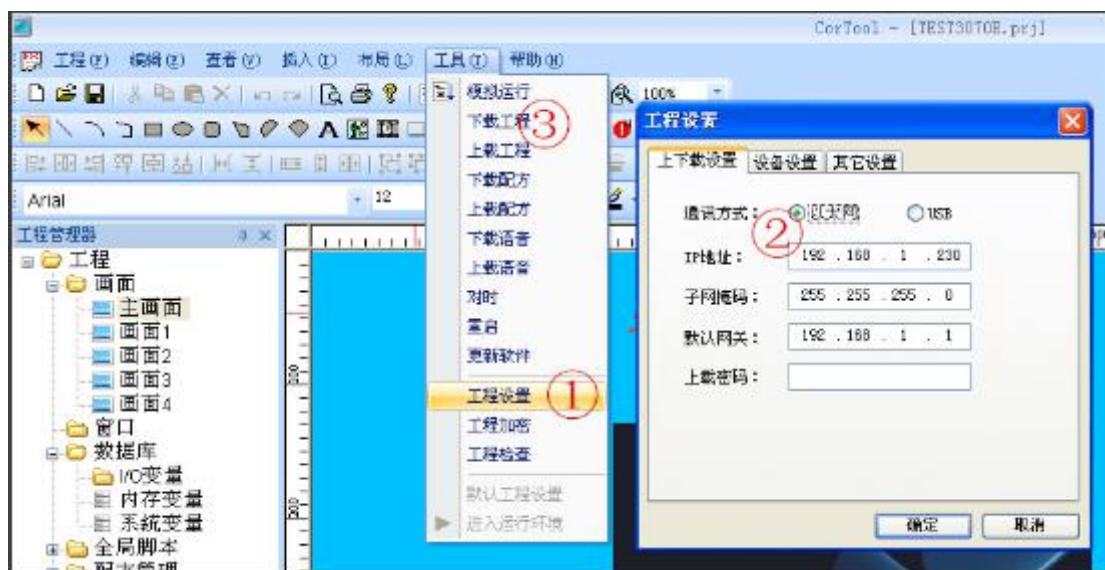
第一步：必须把本地电脑 IP 设置为与 HMI 同一网段的不同地址 例：192.168.1.xx

HMI 的默认 IP：192.168.1.230

子网掩码：255.255.255.0



第二步：打开 corware 组态软件，工程编辑完后，点击菜单栏中“工具”“工程设置”，在弹出框中填写 IP，完成后点击“工具”“工程下载”，进度条走完即下载成功。(若工程内有配方，需点击“工程设置”“下载配方”)



13.1.2 方法二 {U 盘}

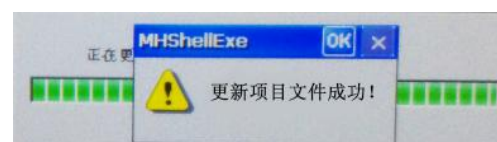
第一步：本地电脑上打开 corware 组态软件，工程编辑完后，点击菜单栏中“保存”按钮。

第二步：在电脑上插上 U 盘，点击菜单栏中“工具”“工程打包”，在 U 盘中自动生成“hmi.prjp”文件。

第三步：将 U 盘连接 TPC，使用电源给 TPC 供电，开机启动后在（如右图）时点击屏幕，进入设置画面。



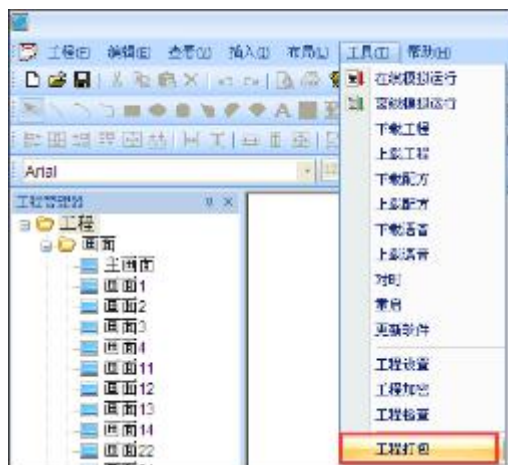
第四步：点击“U 盘更新程序”按钮，在出现“更新项目文件成功”后，点击“OK”。



第五步:点击“重启”按钮，TPC 重启后即进入工程界面，工程更新完成。

13.1.3 方法三 {U 盘}

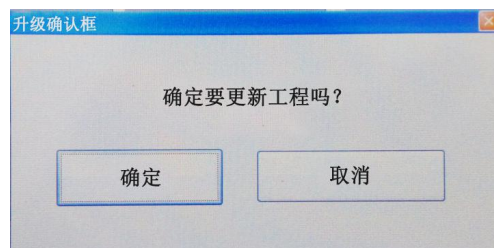
第一步:本地电脑上打开 corware 组态软件，工程编辑完后，点击菜单栏中“保存”按钮。



第二步:在电脑上插上 U 盘，点击组态菜单栏中“工具”“工程打包”，在 U 盘中自动生成“hmi.prjp”文件。



第三步:使用 24V 直流电源给 TPC 供电，进入组态工程界面，插上存有“hmi.prjp”文件的 U 盘，弹出右图对话框，点击“确定”按钮后开始更新工程。



第四步:更新完成后，自动进入工程首页，此时工程更新完毕。

13.2 时间显示和时间修改

1. 时间显示

1.1 文本元件显示：【属性框】【动态属性】【文本显示】，关联系统变量“\$SystemTime”。



第一步，新建一个文本显示框

动态属性	
水平移动	
垂直移动	
宽度变化	
高度变化	
数值显示	
文本显示	\$SystemTime
文字颜色	

第二步，关联系统变量
\$SystemTime

第三步，保存后离线模拟或下载到 HMI 显示

2017-02-23 15:58:52

1.2 数字时钟显示：【菜单栏】【插入】【插件】【数字时钟】【】



属性框	
基本属性	
名称	Obj182
X	554
Y	26
宽度	193
高度	48
可见性	只在本画面中...
数字时钟属性	
设置	...



Now 返回当前的系统日期和时间；

Date 返回当前的系统日期；

Year 返回一个代表年份的数字；

Month 返回一个代表月份的数字；

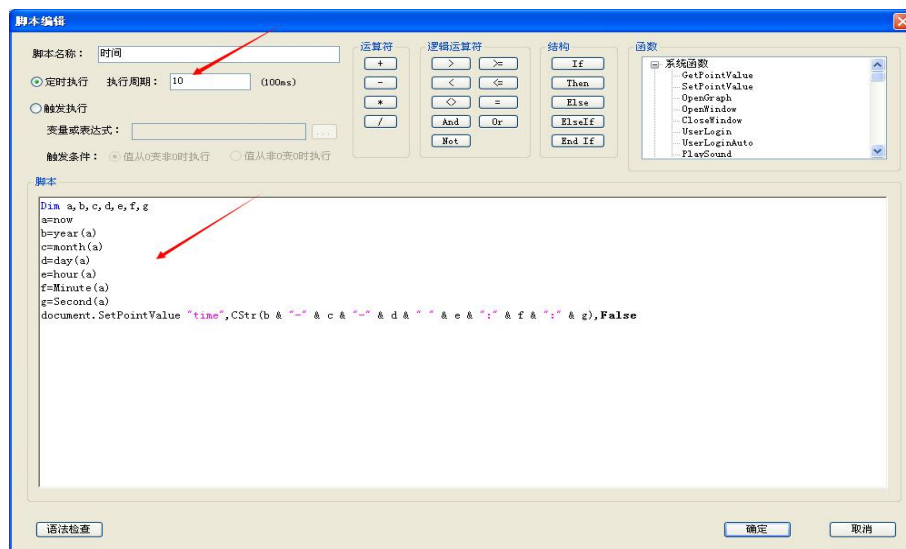
Day 返回一个代表天数的数字；

Hour 返回一个代表小时的数字；

Minute 返回一个代表月份的数字；

Second 返回一个代表秒的数字。

例子：



通过新建一个全局脚本，执行周期 10（单位 100ms），实现日期和时间的在线显示。脚本每 1S 执行一次，如下：

a=now

b=year(a)

```
c=month(a)
```

```
d=day(a)
```

```
e=hour(a)
```

```
f=Minute(a)
```

```
g=Second(a)
```

```
document.SetPointValue "time",CStr(b & "-" & c & "-" & d & " " & e  
& ":" & f & ":" & g),False
```

运行时，将日期和事件赋给字符串变量 time，将 time 通过文本输出实时显示。 2017-01-05 16:44:22

2.时间修改

时间“文本”元件，【属性框】【脚本】【按下时执行】，调用系统函数 SetSystemTime 或输入以下脚本：

```
document.SetSystemTime
```

运行时点击文本弹出时间修改窗口。

注：该修改时间方式只在 HMI 中有效，在 PC 模拟运行时无效。

13.3 多语言

多语言：可自由选择工程字符语言属性，如中文、英文等。

1.通过【属性框】【动态属性】【文本显示】中变量的阈值不同，显示不同的字符串，从而显示不同的语言。



2.将所有文本关联同一个变量，新建按钮，关联变量，通过设置变量状态或弹窗选择不同语言。

13.4 掉电保存

掉电保存：HMI 断电重启后，变量值与断电前相同。

1.方式一：脚本保存

1.1 通过脚本函数“CreateHmiFile”“WriteString”将变量值写入 NANDFLASH 下文本中。

CreateHmiFile

注释：该函数用来创建文件。

例子：document.CreateHmiFile"NANDFLASH\1.txt"

运行时，在 NANDFLASH 文件夹下创建名称为“1”的文本文档。

WriteString

注释：该函数用来写字符串数据。

例子：`document.WriteString"NANDFLASH\1.txt","你好",1`

运行时，向 **NANDFLASH** 文件夹下名称为“1”的文本文件的第 1 行写入字符串“你好”。

1.2 通过脚本函数“ReadString”读取保存在 NANDFLASH 下的变量数值。

ReadString

注释：该函数用来读字符串数据。

例子：`a=document.ReadString("NANDFLASH\1.txt",1)`

`document.SetPointValue "D0",CStr(a),False`

运行时，读取 **NANDFLASH** 文件夹下名为“1”的文本文件的第 1 行的数据，将数据赋给字符串变量 **D0**。

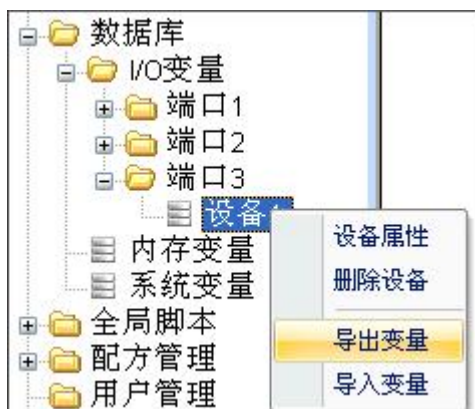
2.方式二：配方保存

配方中的变量值保存在配方文件中，断电重启后变量值不丢失。

13.5 设备变量导入和导出

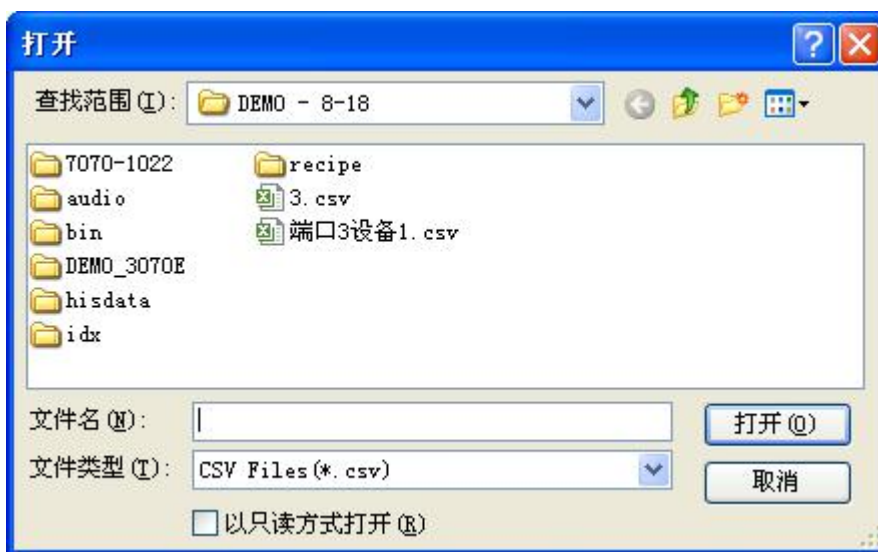
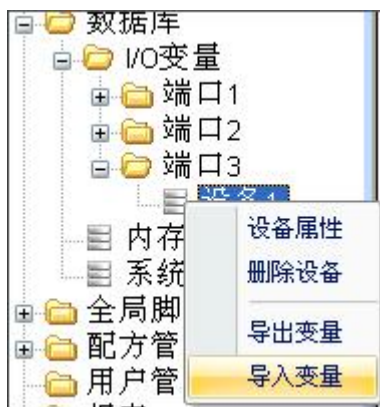
1.设备变量导出

选中【工程管理器】【端口】下【设备】，右键列表中选择【导出变量】，在弹窗中选择存储路径，文件名默认为（端口号+设备号），格式为 CSV。



2. 设备变量导入

选中【工程管理器】【端口】下【设备】，右键列表中选择【导入变量】，在弹窗中选择文件路径，文件名为（端口号+设备号），须与导入的设备端口和设备号一致。



13.6 Modbus 变量转发

Modbus 变量转发功能：从机实时同步主机中变量数值

1.主机设置

1.1 新建串口属性设置如下：

端口属性

端口名称： 端口2 端口类型： 串口

设备厂家： MODBUS 设备参数： ...

设备类型： Modbus_RTU

串口参数

串口号： COM2 波特率： 9600

校验位： 无校验 数据位： 8

停止位： 1 超时时间： 500 ms

以太网参数

IP地址： 192 . 1 . 0 . 3 端口号： 0

确定 取消

1.2 设备属性设置如下，设备地址与从机【从站号】一致：

设备属性

设备名称： 设备1 设备地址： 1

确定 取消

1.3 变量属性设置：

名称	可自行定义
寄存器类型	W4X
数据类型	可选，暂不支持字符串型
寄存器地址	与从机寄存器地址一致

变量属性

基本属性 存盘属性 报警属性 量程变换

名称:

描述:

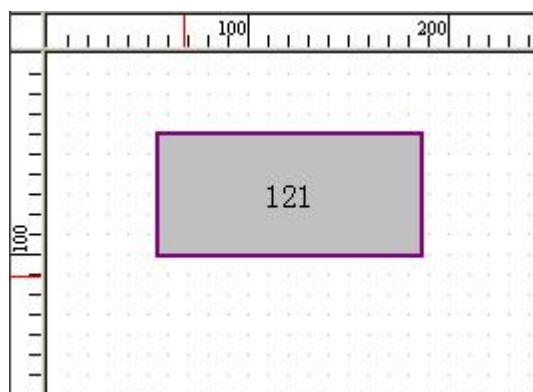
寄存器类型: 数据类型:

数据块块号: 寄存器地址: 数据位:

最小值: 只读 操作记录

最大值: 只写 原始值是码值

1.4 关联变量:



动态属性	
水平移动	
垂直移动	
宽度变化	
高度变化	
数值显示	
文本显示	zf01
文字颜色	
线条颜色	
填充颜色	
可见性	
闪烁	

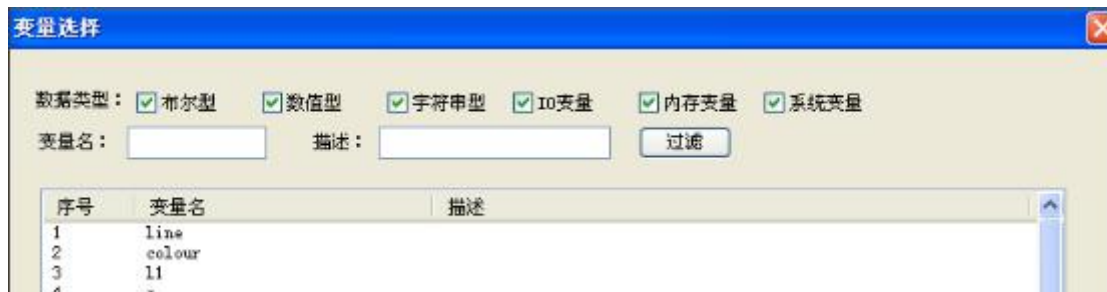
2.从机设置

2.1 点击菜单栏【工程】【工程设置】【Modbus 从站设置】【启用 Modbus 从站】



从站号	设置与主机设备地址一致
波特率	与主机串口波特率一致

2.2 点击【映射表】右侧按钮，点击右侧【增加映射】按钮，选择映射变量，变量的数据类型与主机变量一致，寄存器地址设置与主机变量寄存器地址相同。



2.3 关联变量



2.4 下载工程到 HMI 后，用通讯线连接主机和从机，从机中映射的变量值即与主机同步。

13.7 背光控制

1. 在 Corware 菜单栏【工具】【工程设置】【其它设置】中，设置“背光关闭等待时间”。

设置为 0 表示永不待机，设置为 N 表示 N 分钟无操作后开始待机。

注：产生报警时会复位待机时间



2. 函数设置背光状态。

例子: `document.SetBacklightState(False)`

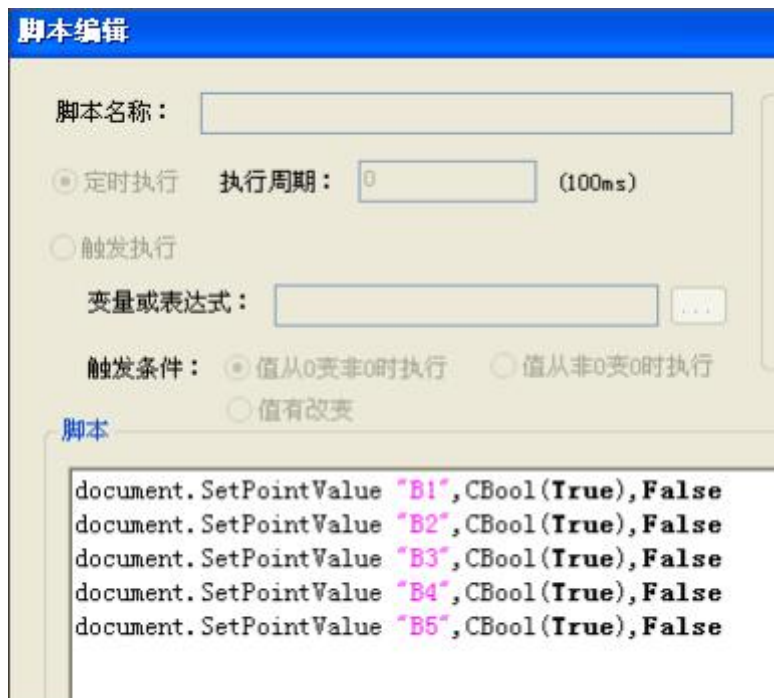
`document.SetBacklightState(True)`

运行时, 设置背光状态打开或者关闭, **False** 为立即进入背光状态再次点击进入画面, **True** 为进入背光等待状态, 与软件中【工具】【其他设置】【背光关闭等待时间】结合使用。

13.8 批量赋值

批量赋值: 通过脚本一次性给多个变量赋值。

例: 在按钮【属性框】【脚本】【按下时执行】输入以下脚本, 运行时点击按钮即可将变量 B1、B2、B3、B4、B5 同时置 1。



13.9 通讯控制

通过函数“SetDevice”来打开或关闭设备驱动。

例子：打开：document.SetDevice "端口 1","设备 1",1

关闭：document.SetDevice "端口 1","设备 1",0

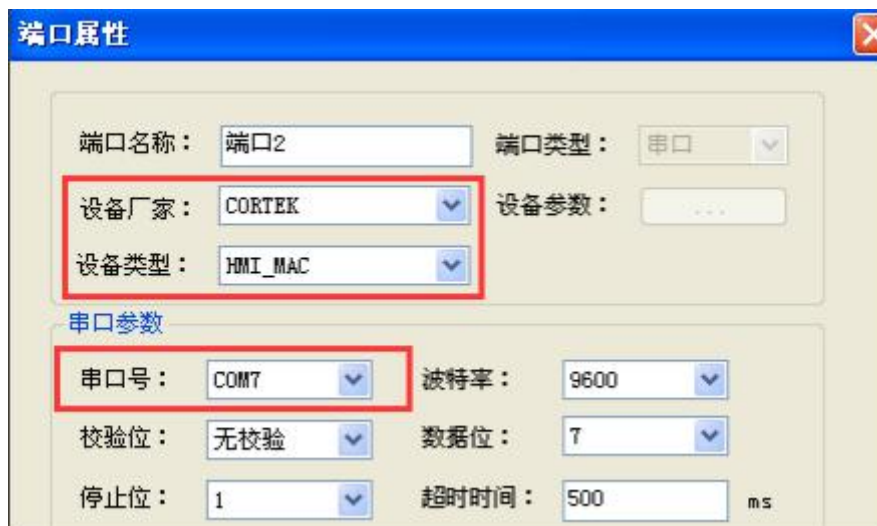
运行时，打开或关闭端口 1 下设备 1 的驱动，1 为打开，0 为关闭。

13.10 MAC 地址读取

MAC（Media Access Control，[介质访问控制](#)）地址，也叫硬件地址，长度是 48 比特（6 字节），由 16 进制的数字组成,例：90：70：16：10：AD：4F

.此处介绍读取 MAC 地址的方式是：通过变量显示寄存器地址中存储的 MAC

1.在【工程管理器】【数据库】【I/O 变量】中新建端口，端口属性中【设备厂家】选择“CORTEK”，【设备类型】选择“HMI_MAC”，【串口号】选择尚未被使用的串口号。



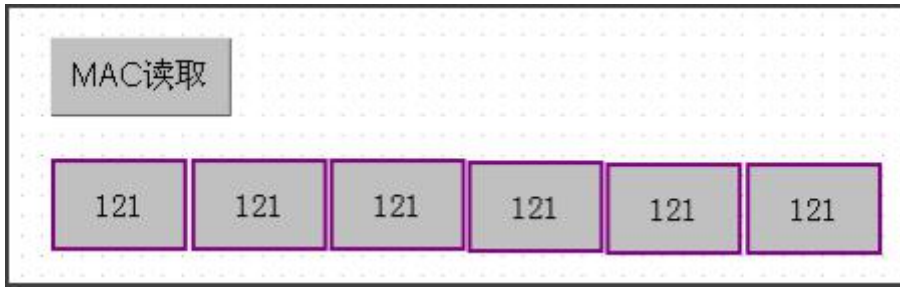
2.右键新建的端口，选择【新建设备】。



3.在编辑区域右键“批量增加变量”，变量个数选择“7”，变量名前缀“MAC”，寄存器类型“MAC”，数据类型“短整型”，寄存器地址“0”，地址间隔“1”。

序号	变量名	寄存器地址	数据类型	读写属性
1	MAC0	MAC0	短整型	读写
2	MAC1	MAC1	短整型	读写
3	MAC2	MAC2	短整型	读写
4	MAC3	MAC3	短整型	读写
5	MAC4	MAC4	短整型	读写
6	MAC5	MAC5	短整型	读写
7	MAC6	MAC6	短整型	读写

4.在画面中放置 1 个按钮和 6 个数据输入框。



5.在按钮【属性框】【操作】【设置数值】，关联变量”MAC0”，点选“键盘输入”。



6.数据输入框【属性框】【动态属性】【数值显示】依次关联变量“MAC 1 ~MAC 6”，显示格式“十六进制”



7.保存工程，下载工程到 HMI 中，点击按钮，键盘输入 1，即可在 6 个数据输入框中显示出 HMI 的 MAC 地址。如果数据输入框中显示的数值只有一位，需在该数字前加 0，或在全局脚本中通过脚本自动加 0。

13.11 码值

码值：用于将下位机整型数据在上位机上显示为浮点型数据

- 1.新建单精度浮点型变量，在变量属性中勾选“原始值是码值”，设置最小值和最大值，与【量程变换】中数值范围构成变换比例，如下图中将 PLC 中原数据缩小 100 倍。

变量属性

基本属性 存盘属性 报警属性 量程变换

名称：

描述：

寄存器类型： 数据类型：

数据块块号： 寄存器地址： 数据位：

最小值： 只读 操作记录

最大值： 只写 原始值是码值

变量属性

基本属性 存盘属性 报警属性 量程变换

进行量程变换

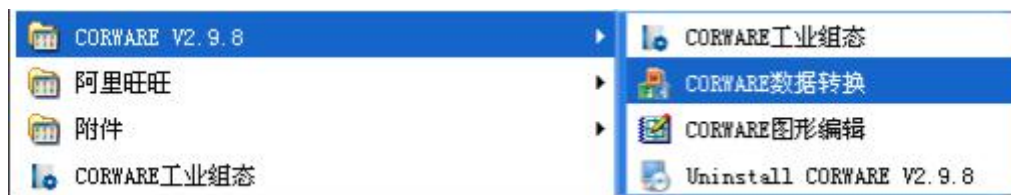
原始最小值 原始最大值

13.12 数据转换

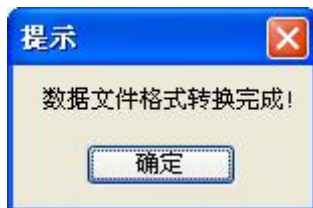
数据转换：将 HMI 中存储的历史数据.dat 文件转换成.CSV 文件

1.

- 1.在 PC【开始】【所有程序】【CORWARE V3.2.0】，右侧菜单中双击【CORWARE 数据转换】，弹出 HisDatetoCSV Tool 窗口。



2. 点击‘单文件转换’或‘文件夹转换’，弹出文件路径窗口，选择.dat文件，点击‘打开’，转换完成弹出完成提示，如下图：



13.13 音视频调用

13.13.1 音频

<音频>调用脚本：`document.PlaySound " ***.wav "`

音频文件路径：`NANDFlash\bin\audio`

<音量+>调用脚本：`document.Volumeincr`

<音量->调用脚本：`document.Volumedecr`

注意：外接 Speaker 参数建议使用 **4Ω 3W** 或 **4Ω 2W**

13.13.2 视频

<视频>调用脚本：`document.VideoPlayer "NANDFlash\Video***.avi"`

视频文件路径：`NANDFlash\Video\`

例：

1.用网线连接 HMI 和本地 PC;

2.HMI 开机进入工程；打开本地 PC “我的电脑”，在地址栏中输入“`ftp://192.168.1.230`”，按 `enter` 键后进入 HMI 存储空间；将准备好的视频文件复制到 HMI 路径：`NANDFlash\Video\Test.avi`

3.在上位机工程画面中新建按钮“视频测试”，下图圈 1 所示；

4.点击软件右下脚属性框 “脚本” “按下时执行”，下图圈 2 所示，在弹出的脚本编辑框中输入：

```
document.VideoPlayer "NANDFlash\Video\Test.avi"
```

5.保存工程，下载到 HMI 中，点击“视频测试”按钮即可播放视频。

6.播放视频会连续循环播放，双击屏幕退出全屏模式，然后点右下角 X 按钮关闭播放器，退出到组态下。



13.14 图符库

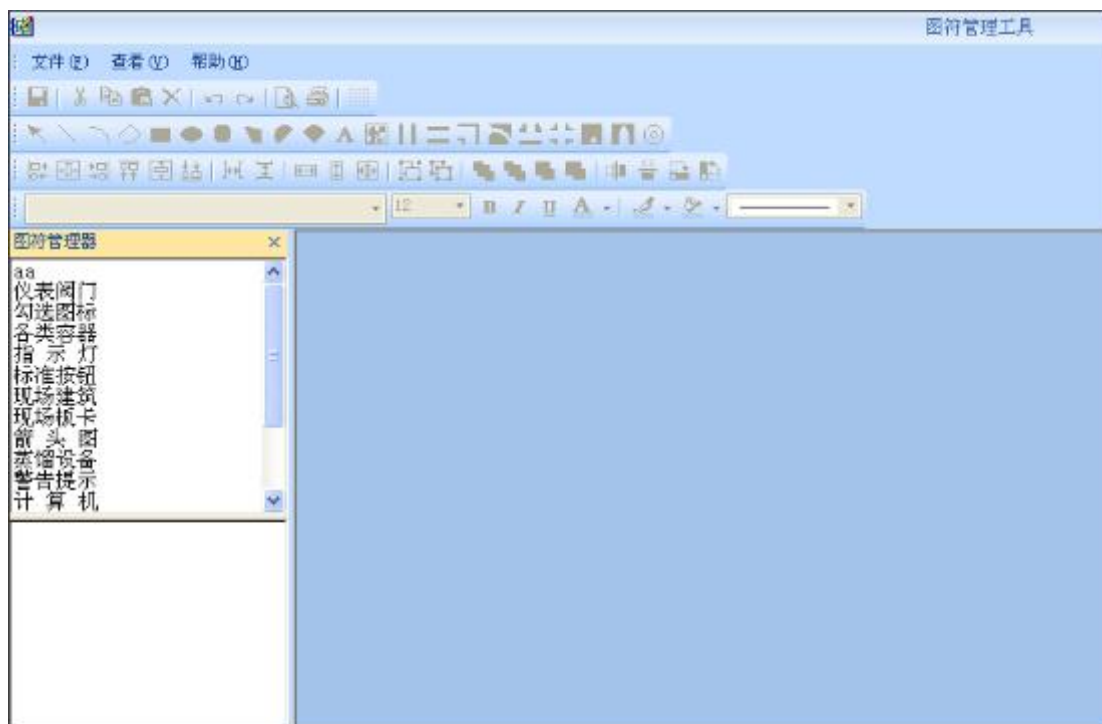
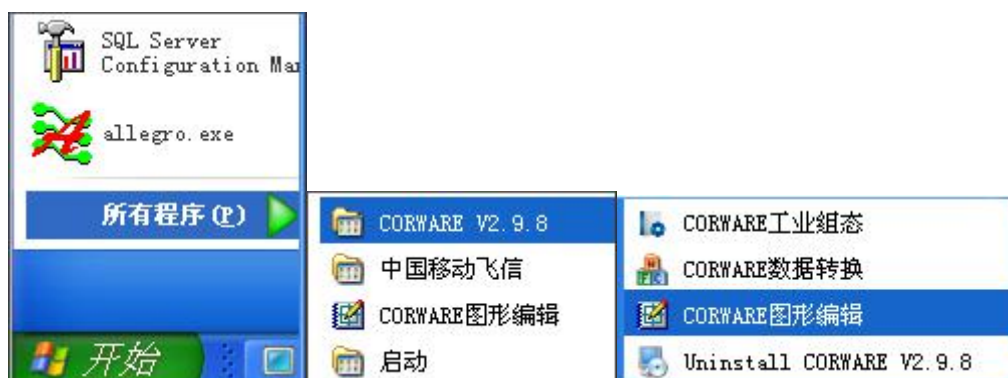
图符库概述

图库是指 CORWARE 中提供的已制作成型的图素组合。图符库中的每个成员称为图符。

使用图库开发工程界面至少有三方面的好处：一是降低了工程人员设计界面的难度，使他们能更加集中精力于维护数据库和增强软件内部的逻辑控制，缩短开发周期；二是用图库开发的软件将具有统一的外观，方便工程人员学习和掌握；最后，利用图库的开放性，工程人员可以生成自己的图库元素，“一次构造，随处使用”，节省了工程人员投资。

CORWARE 为了便于用户更好地使用图库，提供图符管理器。图符管理器集成了图符库管理的操作，在统一的界面上，完成“新建图符”，“编辑图符”，“删除图符”，“图符重命名”等图符操作，“新建图符分类”，“删除图符分类”，“图符分类重命名”等分类操作。

1.在 PC【开始】【所有程序】【CORWARE V2.9.8】，右侧菜单中选择【CORWARE 图形编辑】，即可打开该工具。



3.新建图符

2.1 点击图符分类，在图符分类下方图符列表中右键，选择【新建图符】，



2.2 在‘新建图符’窗口中输入图符名称和状态数目，状态数目表示图层数，运行时可通过变量数值不通显示不同的图符。

13.15 历史文件导出

历史文件包括历史报警文件和历史数据文件，

历史文件导出以函数命令 CopyHisData 方式实现：在 U 盘或 SD 卡根目录新建 hisdata 文件夹，hisdata 文件夹下新建 Alarm 和 Trend 文件夹，分别存放报警和数据文件，文件格式都是 “.dat”。

使用格式转换工具 HisDatetoCSV Tool 可将 “.dat” 文件转换成 Excle 可以正常打开的 CSV 文件

CopyHisData

注释：该函数用来复制历史报警和历史数据文件到 U 盘或 SD 卡。

返回值：无

例子：document.CopyHisData 0, 0, 2016, 5

运行时，复制 2016 年 5 月的历史数据到 U 盘。

document.CopyHisData 参数 1, 参数 2, 参数 3, 参数 4

参数 1：0 为 U 盘，1 为 SD 卡。

参数 2：0 为不删除源文件，1 为删除源文件

参数 3：历史数据年份

参数 4：历史数据月份

注意：该函数只能导出整月的历史数据，运行时用系统变量 \$CopyAlarmPercent 和 \$CopyTrendPercent 显示历史报警和历史数据文件拷贝的百分比。

13.16 插件功能

插件实际上是可重用对象，用来执行专门的任务。每个插件实质上都是一个微型程序，但不是一个独立的应用程序，通过控件的属性、方法等控制控件的外观和行为，接受输入并提供输出。插件与图符相比功能更加有针对性，开发插件需要高级编程语言实现。

外观上类似于组合图素，工程人员只需把它放在画面上，然后配置查件的属性,插件就能完成复杂的功能。当所实现的功能由主程序完成时需要制作很复杂的命令语言，或根本无法完成时，可以采用插件。

内置插件是 CORWAR 提供的插件，无需工程技术人员利用高级语言开发的插件。目前支持“系统时间”、“数字时钟”、“数码管”、“输入框”、“走马灯”、“二维码”6

个插件。点击菜单“图元插件”，即可选择对应的插件了。



1.新建插件：在工具栏中点击插件按钮，弹出“插件”窗口如图，



选择要使用的插件，点击确定，然后再窗口中按着鼠标左键拖动一个矩形后松开鼠标按键，就会在窗口中生成相应的插件。

2019-04-11 12:25:08

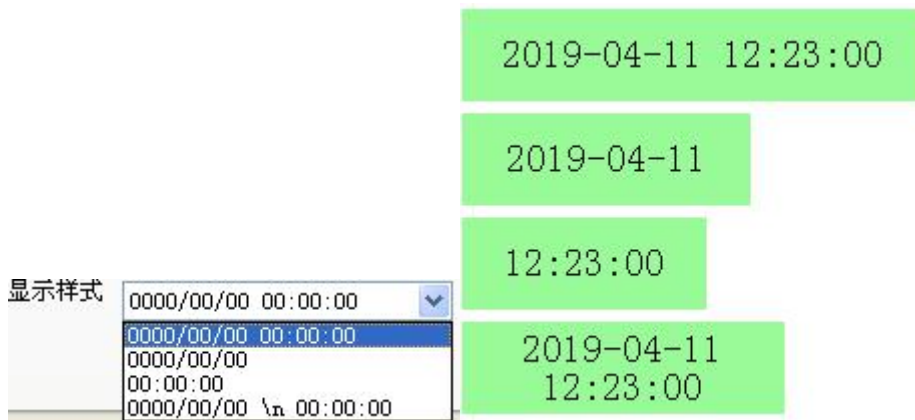
2.系统时间插件：默认样式如图所示，



在属性栏里双击设置，弹出“系统时间属性”窗口，



在窗口中可修改系统时间的字体、字体颜色、背景颜色、显示样式，其中显示样式有如下四种，



3. 数字时钟插件：默认样式如图所示，


12:24:48



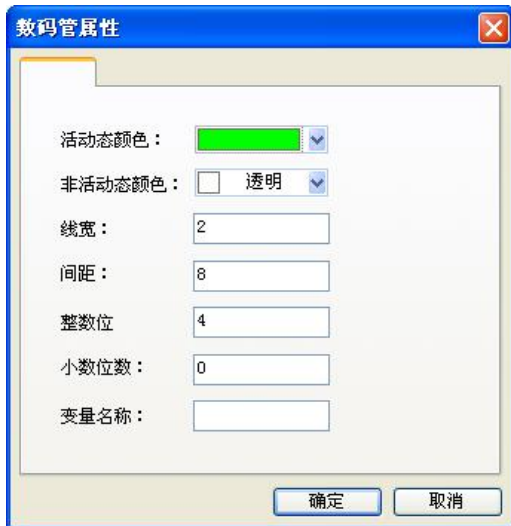
在属性栏里双击设置，弹出“数字时钟属性”窗口，



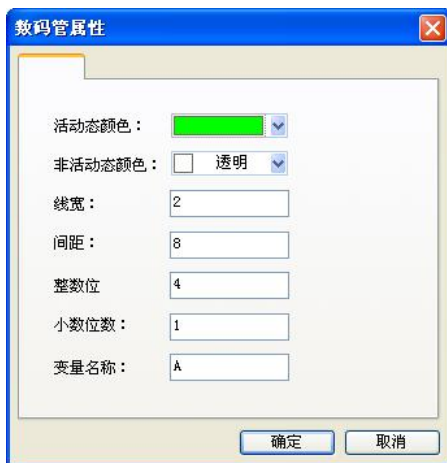
在窗口中可修改数字时钟的颜色、背景色、线宽、间距。

4. 数码管插件：默认样式如图所示，

在属性栏里双击设置 ，弹出“数码管属性”窗口，



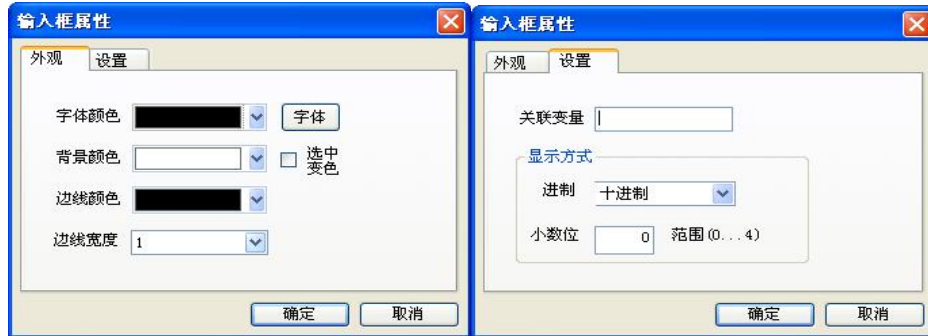
在窗口中可修改数码管的颜色、背景色、线宽、间距、整数位、小数位，但是要输入变量名称才能显示数字，变量需要是数值类型的（整型、浮点型）。例如：变量 A 为单精度浮点型，设置小数点位数为 1 位，显示效果如下，



5. 输入框插件：默认样式如图所示，

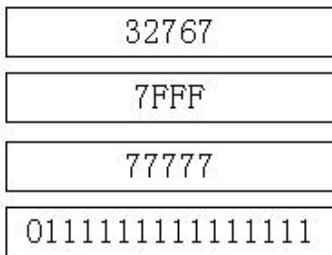


在属性栏里双击设置 **输入框属性** 设置，弹出“输入框属性”窗口，

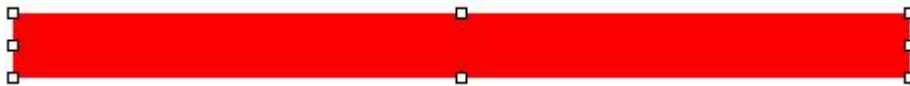


在外观页中可修改输入框的字体颜色、背景颜色、边线颜色、边线宽度。

在设置页中要输入关联的变量名，变量需要是数值类型的（整型、浮点型），在显示方式中可设置不同的进制（十进制、十六进制、八进制、二进制）以及小数点位数，例如：变量 B 为短整型，设置不同的进制后显示效果如下图，点击输入框插件可弹出输入键盘。



6. 走马灯插件：默认样式如图所示，



在属性栏里双击设置 **走马灯属性** 设置，弹出“走马灯属性”窗口，



在窗口中可修改走马灯的字体颜色、背景颜色、滚动速度（默认每个 0.5s 滚动一次，一次滚动 2 个字）。走马灯插件的作用是有报警的情况下滚动显示报警信息，例如：设置变量 B 报警属性为上限报警，报警值为 50，当 B 的值大于 50 时，走马灯就会滚动显示变量 B 的

报警信息。



51

变量B高限报警 变量B高限报警 变量B高限报警

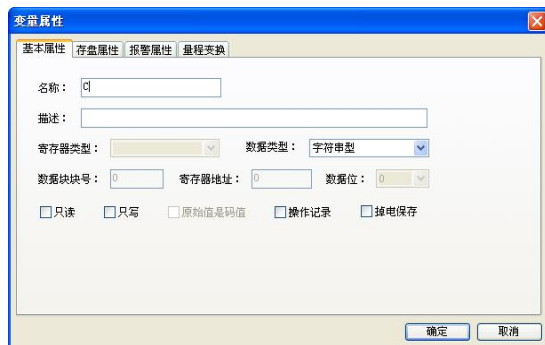
7. 二维码插件：默认样式如图所示



在属性栏里双击设置 **二维码属性** 设置 ... ，弹出“二维码属性”窗口，



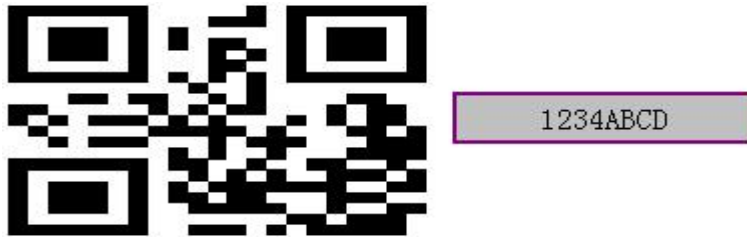
在窗口中输入关联变量的名称，此变量必须是字符串型变量；
例如：变量 C 为字符串类型，



二维码属性关联变量 C，



运行中修改 C 的值，二维码插件就会生成相应的二维码图片。



第十四章 工程测试


新建工程在 CORWARE 组态环境中完成(或部分完成)组态配置后,应当转入 CORWARE 模拟运行环境,通过试运行,进行综合性测试检查。

鼠标单击菜单“工具”中的“模拟运行”按钮,或操作快捷键 Ctrl+F5,即可进行模拟运行,在模拟环境中对于要实现的功能进行测试。

在组态过程中,可随时进入运行环境,完成一部分测试一部分,发现错误及时修改。主要从以下几个方面对新工程进行测试检查:外部设备、动画动作、按钮动作、图形界面、运行脚本。

14.1 组态检查

工程组态到一定阶段后,可以先在编辑环境中进行工程检查。

点击菜单“工具\工程检查”,或者常用工具栏的 ,如果有一般性错误,就会提示

技术人员，哪里出现错误。

14.2 外部设备的测试

外部设备是应用系统操作的主要对象，是通过配置在设备窗口内的设备构件实施测量与控制的。因此，在系统联机运行之前，应首先对外部设备本身和组态配置结果进行测试检查。

首先确保外部设备能正常工作，对硬件设置、供电系统、信号传输、接线接地等各个环节，先进行正确性检查及功能测试，设备正常后再联机运行。

其次在数据库组态中，要反复检查端口的选择及其属性设置是否正确，设备地址与实际设备地址是否一致，确认正确无误后方可转入联机运行。

联机运行时，首先给外部设备输入标准信号，观察采集进来的数据是否正确，外部设备在手动信号控制下，能否迅速响应，运行工况是否正常等等。

14.3 动画动作的测试

图形对象的动画动作是实时数据库中数据对象驱动的结果，因此，该项测试是对整个系统进行的综合性检查。通过对图形对象动画动作的实际观测，检查与实时数据库建立的关系是否正确，动画效果是否符合实际情况，验证画面设计与组态配置的正确性及合理性。

动画动作的测试建议分两步进行：

首先利用模拟设备产生的数据进行测试，定义若干个测试专用的数据对象，并设定一组典型数值或在运行策略中模拟对象值的变化，测试图形对象的动画动作是否符合设计意图；然后，进行运行过程中的实时数据测试。可设置一些辅助动画，显示关键数据的值，测试图形对象的动画动作是否符合实际情况。

14.4 按钮动作的测试

首先检查按钮标签文字是否正确。实际操作按钮，测试系统对按钮动作的响应是否符合设计意图，是否满足实际操作的需要。当设有快捷键时，应检查与系统其它部分的快捷键设置是否冲突。

14.5 图形界面的测试

图形界面由多个画面及窗口构成,每个画面或窗口由很多图形对象组成,各个图元外观、大小及相互之间的位置关系需要仔细调整和精确定位,才能获得满意的显示效果。在系统综合测试阶段,建议先进行简单布局,重点检查图形界面的实用性及可操作性。待整个应用系统基本完成调试后,再对所有图元的大小及位置关系进行精细地调整。

14.6 运行脚本的测试

应用系统的运行脚本在后台执行,其主要的职责是对系统的运行脚本实施有效控制和调度。运行脚本本身的正确性难于直接测试,只能从系统运行的状态和反馈信息加以判断分析。建议用户一次只对一个脚本进行测试,测试的方法是创建辅助的用户窗口,用来显示策略块中所用到的数据对象的数值。测试过程中,可以人为地设置某些控制条件,观察系统运行流程的执行情况,对脚本的正确性作出判断。同时,还要注意观察脚本运行中系统其它部分的工作状态,检查脚本的调度和操作职能是否正确实施。例如,脚本中要求打开或关闭的窗口,是否及时打开或关闭,外部设备是否按照脚本中设定的控制条件正常工作。