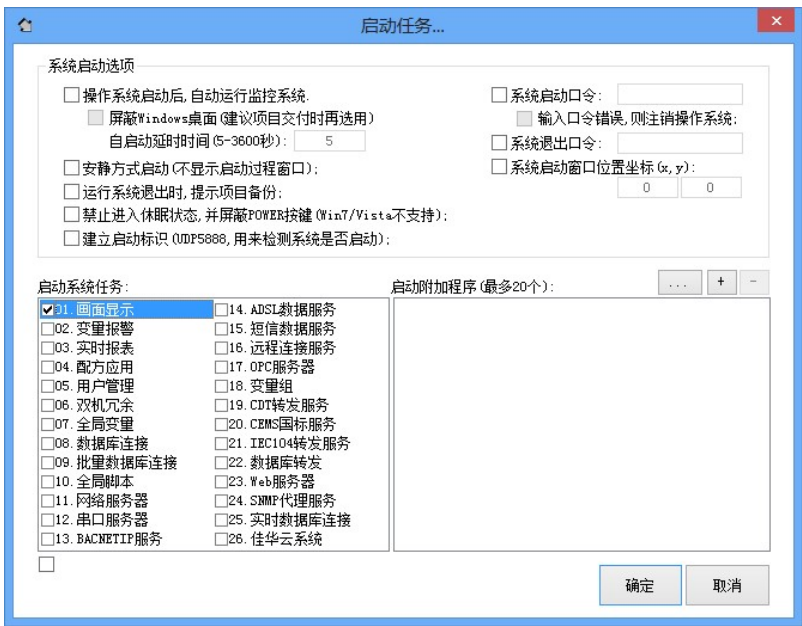


## 9. 画面显示

序号	内容	页码
9.1	自动启动画面显示	9-2
9.2	画面显示属性	9-2
9.3	多窗口显示	9-4
9.4	多屏显示(大画面)	9-5
9.5	多屏显示(多画面)	9-5
9.6	画面漫游	9-6
9.7	通过命令行启动画面显示程序	9-6
9.8	通过编程在其他软件中嵌入画面子窗口	9-7
9.9	画面支持无级缩放	9-11
9.10	加锁和解锁显示画面	9-11
9.11	画面支持滑动功能	9-12
9.12	画面回放(CSV 文件)	9-12
9.13	画面回放(实时数据库)	9-17
9.14	根据报警状态制作故障诊断	9-19
9.15	实现设备维护提醒画面	9-22
9.16	拷贝画面内容到 Word 文档	9-27
9.17	画面某段时间无操作切换到其他画面	9-28
9.18	通过文本文件配置自由表格	9-29
9.19	模糊查找画面对象	9-31
9.20	变量无效动态颜色指示	9-33
9.21	动态修改树形框 XML 文件	9-34
9.22	获取画面键盘输入内容	9-36
9.23	设备沿运行轨迹移动	9-37
9.24	嵌入百度地图	9-41

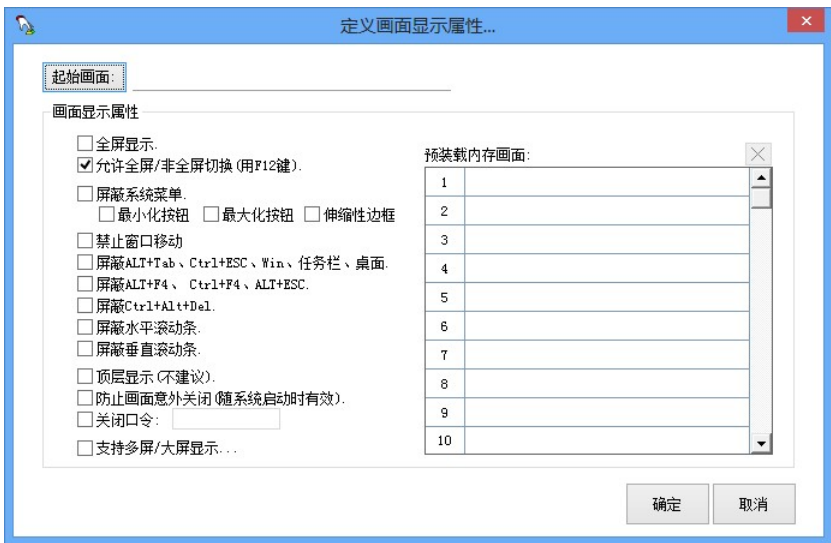
### 9.1 自动启动画面显示

- 画面显示随系统启动, 需要在启动任务中选择;
- 选择执行[我的系统. 设置. 启动任务]功能, 选择启动画面显示:



### 9.2 画面显示属性

- 选择执行[画面显示. 显示属性]:



- 通过[起始画面]按钮, 选择缺省显示画面, 未设定起始画面, 则启动时弹出画面列表手动选择;
- 选择[全屏显示], 则可以屏蔽掉显示画面窗口的标题、菜单、工具条、状态栏等, 整个显示器屏幕只显示画面内容, 如果画面的尺寸大于屏幕尺寸, 会出现滚动条来漫游屏幕;
- 如果选择“允许全屏/非全屏切换(用F12键)”, 则可以用键盘F12键, 使画面显示窗口在全屏/非全屏之间进行切换; 如果不选择此选项, 则画面显示窗口被全屏显示后, 不能够再被恢复为非全屏;

通过脚本函数可以在全屏与非全屏间切换：

`Picture.SwitchFullScreen` '全屏/非全屏切换；

- ☐ 屏蔽系统菜单, 防止在非全屏模式下, 通过系统菜单操作画面；
- ☐ 显示最小化按钮, 系统菜单未被屏蔽时, 在画面右上角显示最小化按钮；
- ☐ 防止操作者通过热键切换到其他窗口, 应该选择“屏蔽 ALT+Tab、Ctrl+ESC、任务栏”；
- ☐ 防止操作者通过热键关闭窗口, 应该选择“屏蔽 ALT+F4、Ctrl+F4、ALT+Esc”；
- ☐ 防止操作者进行系统操作, 应该选择“屏蔽 Ctrl+Alt+Del”；
- ☐ 屏蔽垂直/水平滚动条, 当画面尺寸与画面窗口不匹配而出现滚动条时, 可不显示滚动条；
- ☐ 要防止其他应用程序窗口覆盖画面显示窗口, 应该选择“顶层窗口, 不被其他窗口覆盖”；  
即使启动了其他应用程序, 画面也总显示屏幕的最前面；  
但也有可能系统本身的某些窗口也不能正常显示, 所以只建议在独立画面时使用；
- ☐ 选择“防止意外关闭”, 可防止画面显示程序被非法关闭；  
即使强制关闭画面显示程序, 系统也会再 15 秒钟内自动启动；  
此功能只在画面显示程序随系统自动启动时有效；
- ☐ 如果选择“关闭口令”, 则能够使画面显示程序被非法关闭时, 出现口令输入窗口, 只有正确输入口令, 才能关闭画面显示, 多用来防止画面显示被非法关闭；
- ☐ 如果希望同时显示多幅画面窗口, 则要选择“手动调整/多画面显示”, 出现下面界面, 并填写各画面的名称、位置、尺寸：

设置多屏或大屏显示...

窗口编号	初始画面	窗口位置 [x, y]	窗口尺寸 [cx*cy]
0	PIC1. drw	0, 0	1366 * 768
1	PIC2. drw	1366, 0	1366 * 768
2			
3			
4			
5			
6			
7			
8			
9			
10			

×

确定

取消

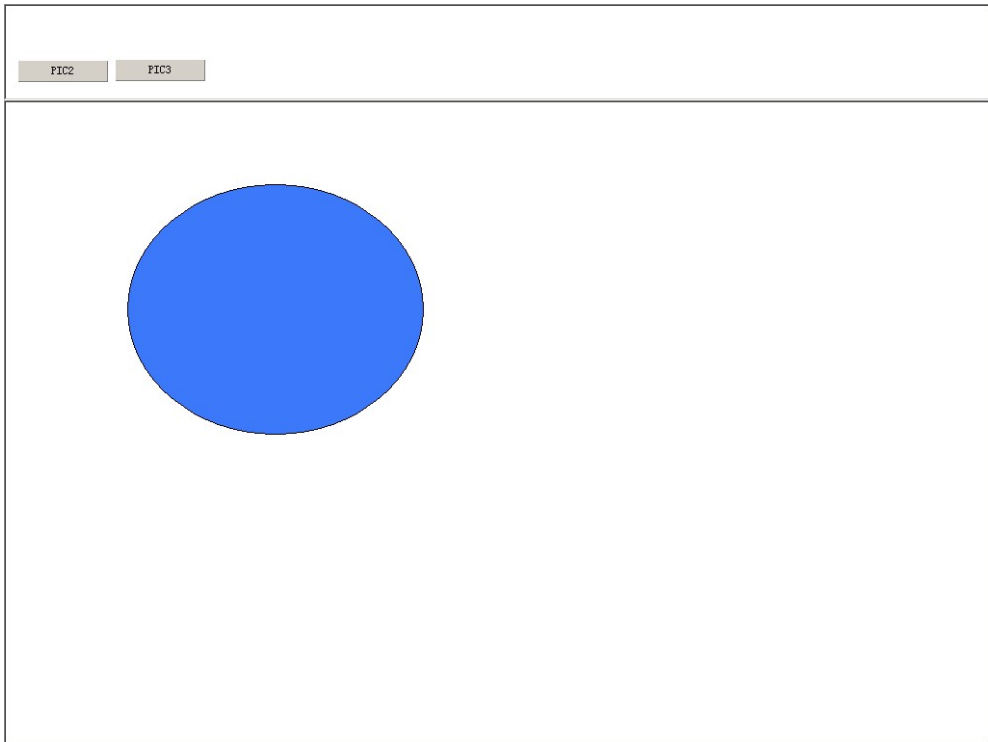
- ☐ 如果希望常用画面之间的切换速度快些, 或者希望系统更稳定一些, 可以把某些经常要打开画面、包含实时曲线的画面, 预先装载到内存中；  
预装载到内存中的画面在切换时不会被关闭, 而是被暂存到某内存区域中；

9.3 多窗口显示

- 制作三幅画面, 假设是 PIC1、PIC2、PIC3;
- 如果显示器分辨率为 1024×768, 则设置 PIC1 尺寸 1020\*96, PIC2、PIC3 尺寸 1020×664;
- PIC1 中做两个按钮, 标题分别为“PIC2”、“PIC3”, 并分别调用脚本:  
`Picture.SwitchMulScrPicture 1, "pic2. drw"`  
`Picture.SwitchMulScrPicture 1, "pic3. drw"`
- 画面显示属性选择[全屏显示]和[手动调整/多画面显示]:



- 运行时, 画面显示如下 (通过 PIC1 中的 PIC2、PIC3 按钮可以切换窗口内容):



PIC1 显示区域(0, 0, 1024, 100), PIC2、PIC3 显示区域(0, 100, 1024, 668);

9.4 多屏显示(大画面)

- 通过分屏卡和多显卡可以使计算机支持多个显示器, 显示某幅大画面;  
假设当前计算机支持两个显示器, 分辨率都是 1024×768, 并设置为水平扩展方式;  
需要制作一幅大画面 PIC1, 水平显示在两个显示器上;
- 组态过程:
  - [1]. 设置画面 PIC1 尺寸 2048×768;
  - [2]. 画面显示属性选择[全屏显示]和[手动调整/多画面显示]:

设置多屏或大屏显示...

窗口编号	初始画面	窗口位置 [x, y]	窗口尺寸 [cx*cy]
0	PIC1. drw	0, 0	2048 * 768
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

确定

取消

9.5 多屏显示(多画面)

- 通过分屏卡和多显卡可以使计算机支持多个显示器, 每个显示器显示不同画面;  
假设当前计算机支持两个显示器, 分辨率都是 1024×768, 并设置为水平扩展方式;  
制作两幅画面 PIC1、PIC2, 分别显示在两个显示器上;
- 组态过程:
  - [1]. 设置画面 PIC1、PIC2 尺寸 1024×768;
  - [2]. 画面显示属性选择[全屏显示]和[手动调整/多画面显示]:

设置多屏或大屏显示...

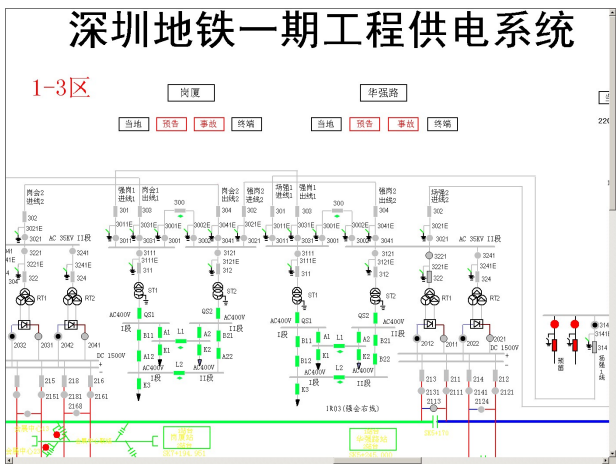
窗口编号	初始画面	窗口位置 [x, y]	窗口尺寸 [cx*cy]
0	PIC1. drw	0, 0	1024 * 768
1	PIC2. drw	1024, 0	1024 * 768
2			
3			
4			
5			
6			
7			
8			
9			
10			

确定

取消

9.6 画面漫游

- 通过漫游方式浏览一幅大画面；  
制作一幅画面 PIC1, 尺寸是 9210×2300, 相当于 27 个 1024×768 个显示器大小；
- 显示结果(通过右侧和下方的滚动条实现漫游显示)：

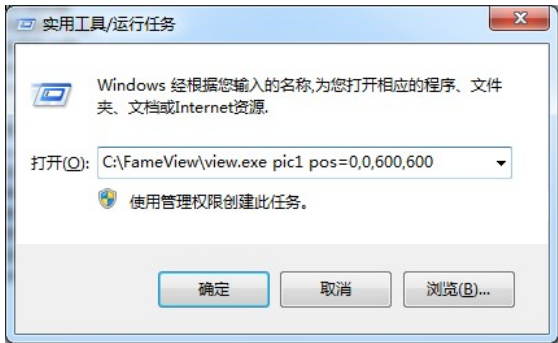


9.7 通过命令行启动画面显示程序

- 通过其他程序或命令行启动画面显示程序, 格式如下:  
AppPath\View.exe filename pos=x,y,sx,sy fullscr=n topwnd=n disablehscroll=n

AppPath	组态软件路径
View.exe	画面显示程序名称
filename	缺省画面名称, 不包括文件后缀
pos=x, y, sx, sy	画面显示位置和尺寸
fullscr=n	可选参数, 是否全屏显示, n=1 全屏显示
topwnd=n	可选参数, 是否顶层显示, n=1 顶层显示
disablehscroll=n	可选参数, 是否屏蔽水平滚动条, n=1 屏蔽水平滚动条
disablevscroll=n	可选参数, 是否屏蔽垂直滚动条, n=1 屏蔽垂直滚动条
MGroupObj=P1, P2	可选参数, 是否支持监控对象, P1(原始对象), P2(实际对象)

- 例如：



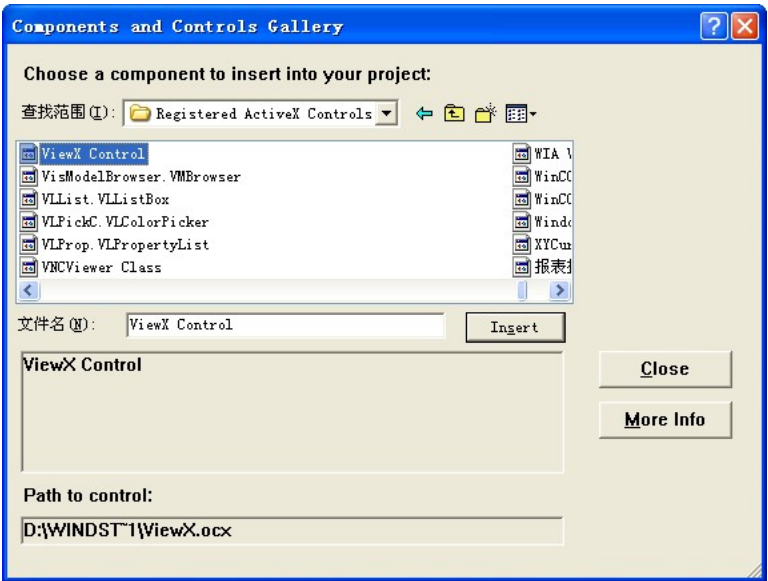
9.8 通过编程在其他软件中嵌入画面子窗口

- 提供画面子窗口控件 (ViewX.ocx), 通过编程可把此控件嵌入到其他系统, 显示动态画面;
- 控件属性及方法:

属性	PictureName	初始画面名称
方法	BOOL OpenPicture()	打开画面并启动控件显示
	Void StopRun()	停止控件并退出工作状态
	BOOL GetExitStatus()	检查控件是否退出工作状态

- 以 VC60 编程为例, 建立对话框程序 (MyView):

[1]. 插入 ViewX 控件到项目中:



[2]. 窗口类中包含头文件 viewx.h, 定义控件变量:

```
#include "viewx.h"

class CMyViewDlg : public CDialog
{
public:
    CViewX* m_MyView;
```

[3]. 窗口类的构造函数, 初始化控件变量:

```
CMyViewDlg::CMyViewDlg(CWnd* pParent /*=NULL*/)
: CDialog(CMyViewDlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CMyViewDlg)
        // NOTE: the ClassWizard will add member initialization here
    //}}AFX_DATA_INIT
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
    m_MyView=NULL;
}
```

[4]. 初始化函数建立控件:

```

BOOL CMyViewDlg::OnInitDialog()
{
    CDialog::OnInitDialog();
    SetIcon(m_hIcon, TRUE);
    SetIcon(m_hIcon, FALSE);
    //建立子窗口控件
    CRect rt;
    GetWindowRect(rt);
    m_MyView=new CViewX();
    m_MyView->Create(NULL,WS_CHILD|WS_VISIBLE, rt,this,18888);
    //设置初始画面
    m_MyView->SetPictureName("c:\\fameview\\PictureFile\\PIC1.drw");
    //可以支持监控对象
    m_MyView->SetPictureName("###GRPMODE1:"+ "P1");
    m_MyView->SetPictureName("###GRPMODE2:"+ "P2");
    m_MyView->SetPictureName("###GRPMODE12");
    //启动画面显示
    if(m_MyView->OpenPicture()==FALSE){
        delete m_MyView;
        m_MyView=NULL;
        return FALSE;
    }
    //根据子窗口标题设置本窗口标题
    CString s=m_MyView->GetTitle();
    if(s.GetLength()>0){
        SetWindowText(s);
    }
    return TRUE;
}

```

[5]. 窗口尺寸变化时, 控件大小也可以变化:

```

void CMyViewDlg::OnSize(UINT nType, int cx, int cy)
{
    if(::IsWindow(m_hWnd)==FALSE) return;
    CDialog::OnSize(nType, cx, cy);
    if(m_MyView==NULL) return;
    CRect rt(0,0,cx,cy);
    m_MyView->MoveWindow(&rt);
}

```



[6]. 如果控件中使用了 Picture.CloseSubWindow 函数来关闭, 则需在窗口中加入:

```
void CMYViewDlg::OnTimer(UINT nIDEvent)
{
    if(nIDEvent==59018) {
        KillTimer(nIDEvent);
        SendMessage(WM_CLOSE, 0, 0);
        return;
    }
    CDialog::OnTimer(nIDEvent);
}
```

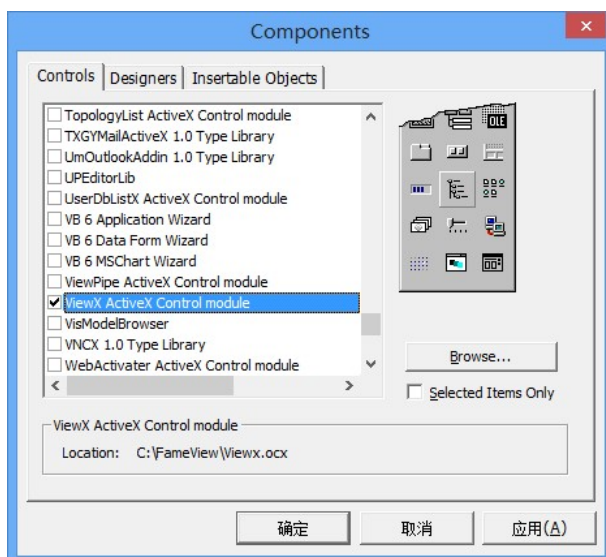
[7]. 窗口关闭时, 关闭控件:

```
void CMYViewDlg::OnDestroy()
{
    if(m_MyView!=NULL) {
        m_MyView->StopRun();
        m_MyView->GetExitStatus();
        delete m_MyView;
        m_MyView=NULL;
    }
    CDialog::OnDestroy();
}
```

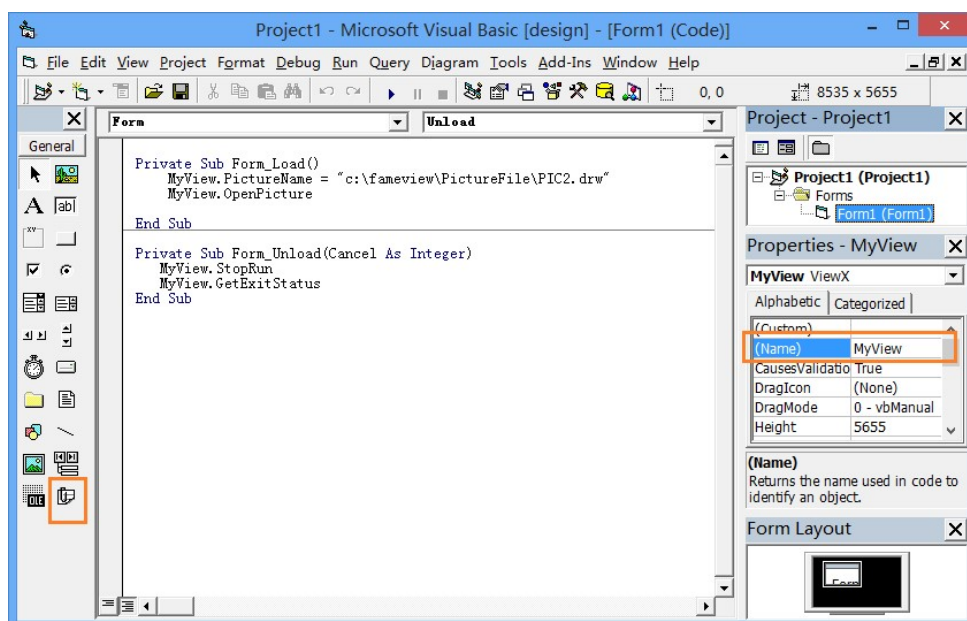
[8]. 程序源代码(MyView.rar)在组态软件工作目录的<Appendix>子目录中;

□ 以 VB6 为例:

[1]. 项目添加组件(ViewX):



[2]. 窗口中添加 ViewX 控件, 并命名为 MyView;



[3]. 窗口打开时, 初始化并启动控件:

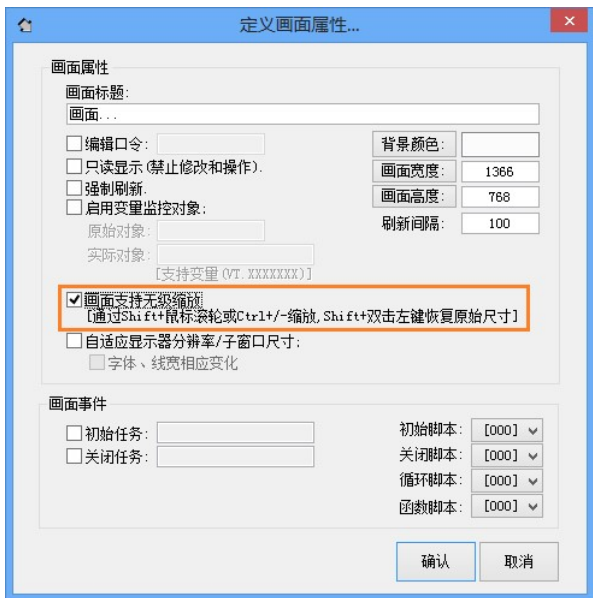
```
Private Sub Form_Load()
    MyView.PictureName = "c:\fameview\PictureFile\PIC2.drw"
    MyView.OpenPicture
End Sub
```

[4]. 窗口关闭时, 使控件停止工作并退出:

```
Private Sub Form_Unload(Cancel As Integer)
    MyView.StopRun
    MyView.GetExitStatus
End Sub
```

## 9.9 画面支持无级缩放

- 制作画面, 设置画面属性, 使此画面具有无级伸缩特性:



- 显示画面时, 缺省按下 Shift 键, 通过鼠标滚轮, 以鼠标所在位置为中心进行缩放;
- 通过脚本实现缩放:

Picture.ZoomPicture m, n, b

m为水平伸缩比例, n为垂直伸缩比例, b=1时字体和线宽随比例伸缩;

放大10%: Picture.ZoomPicture 1.1, 1.1, 1

缩小10%: Picture.ZoomPicture 0.9, 0.9, 1

## 9.10 加锁和解锁显示画面

- 目的: 像对手机键盘进行加锁和解锁一样, 通过组合热键对显示画面的鼠标操作进行加锁和解锁, 避免操作人员离开后, 其他人员非法操作;

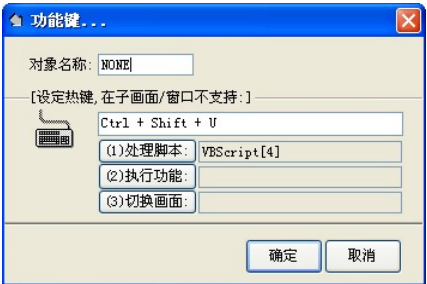
- 通过画面脚本提供的 Picture.LockView 函数实现, 例如:

[1]. 画面增加功能热键, 使用 Ctrl+Shift+L 加锁:



处理脚本内容: Picture.LockView 1

[2]. 画面增加功能热键, 使用 Ctrl+Shift+U 解锁:



处理脚本内容: Picture.LockView 0

### 9.11 画面支持滑动功能

□ 画面显示时执行脚本, 使带有滚动条的主画面支持鼠标或触摸滑动功能:

' [FirstRun]

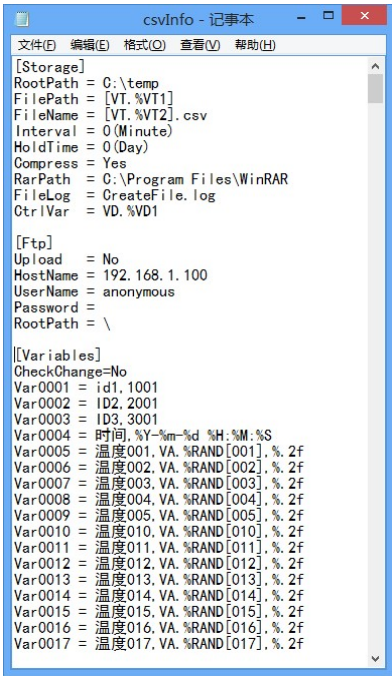
Picture.EnableWindowSliding 1

### 9.12 画面回放 (CSV 文件)

需求:

- [1]. 600 多变量;
- [2]. 某控制变量=1, 秒频率存储到变量值到指定 csv 文件;
- [3]. 某控制变量=0, 停止存储并把当前 csv 文件压缩成 rar 文件;
- [4]. 选择某 rar 文件导入进某 SQL Server 数据表, 根据 SQL Server 数据表内容进行画面回放;

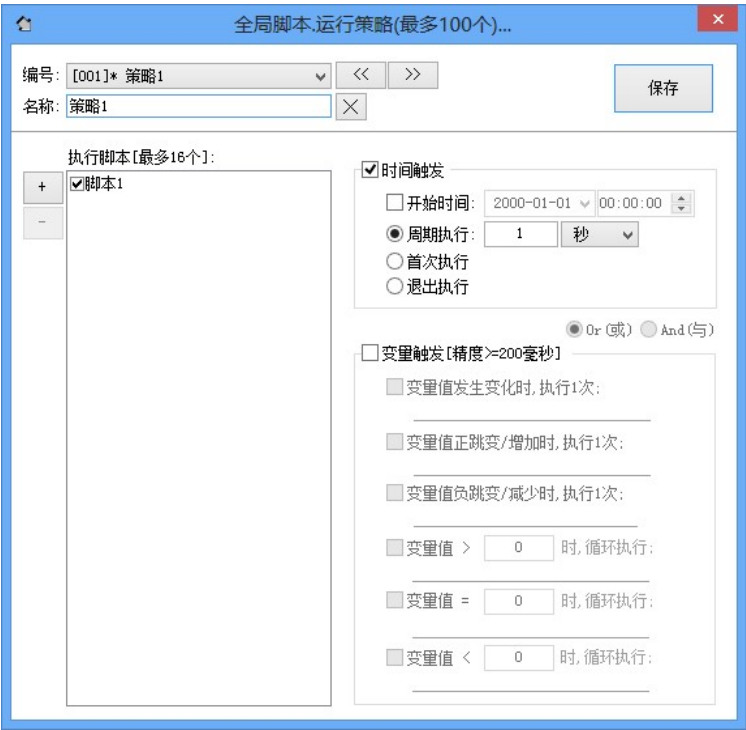
□ Myfile 目录下编写配置文件, 定义存储参数及变量, 如 csvInfo.txt:



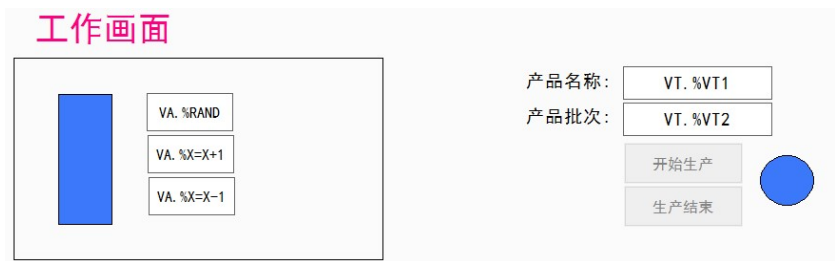
□ 编写全局脚本文件：



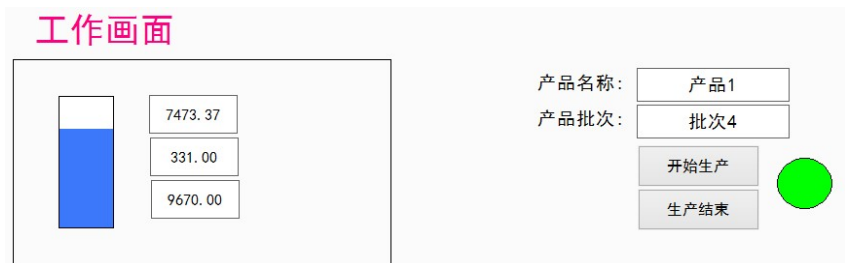
□ 定义运行策略, 每秒运行 1 次脚本文件：



□ 制作测试工作画面：



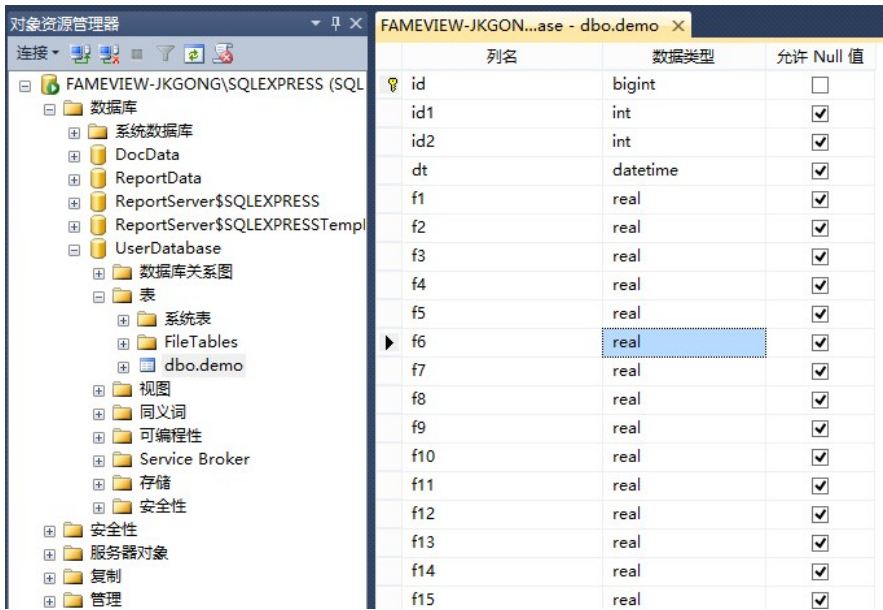
□ 运行工作画面：



输入产品和批次名称并启动后, 则开始在制定目录下生成 csv 文件：



□ 在 SQL Server 中 Userdatabase 中建立 700 个字段的数据表, 考虑导入速度, 不建索引;:



□ 通过数据库连接对应内部变量到字段值：

数据库连接 - [连接1]

连接(F) 窗口(W)

连接方式

☐追加记录.

保存时间:  天 [为0则永久保存]

☒读取记录.

记录号变量:

☐修改记录.

☐存储过程.

控制连接...

时间触发

☐起始时间: 2014-01-01 00:00:00

触发周期: 0 秒

事件触发

[方法1]. 执行38号系统功能 (FB38)

[方法2]. 执行脚本函数 (Runsys.RunDatabaseLink)

最小触发间隔[秒]: 0

序号	字段/参数	类型	对应变量	公式
1	dt	Date	VT.VT3	当前值
2	f1	Real	VA.VA1	当前值
3	f2	Real	VA.VA2	当前值
4	f3	Real	VA.VA3	当前值
5	f4	Real	VA.VA4	当前值
6	f5	Real	VA.VA5	当前值
7	f6	Real	VA.VA6	当前值
8	f7	Real	VA.VA7	当前值
9	f8	Real	VA.VA8	当前值
10	f9	Real	VA.VA9	当前值
11	f10	Real	VA.VA10	当前值
12	f11	Real	VA.VA11	当前值

连接数据库

数据源: FameView UserData Source

用户名:

☐ 令:

数据表: demo

排序字段:

字段与变量对应 (支持500个字段)

字段名称:

类型/处理:

对应变量:

数值公式: 0 - 当前值 采样参数

添加

修改

删除



□ 制作回放画面：

画面回放

VA.VA1

VA.VA2

VA.VA3



调入

播放

快进

停止

□ [调入]按钮执行脚本, 选择 rar 文件并插入内容到数据表：

```
dbName="FameView UserData Source"
tableName="demo"
mainPath="c:\temp"
subPath=""
fileName=""
fileStyle="rar"
rarPath="c:\Program Files\WinRar"
delFile="Yes"
```

9-15

```
clrTable="Yes"
```

```
endVariable="VD.%VD2"
```

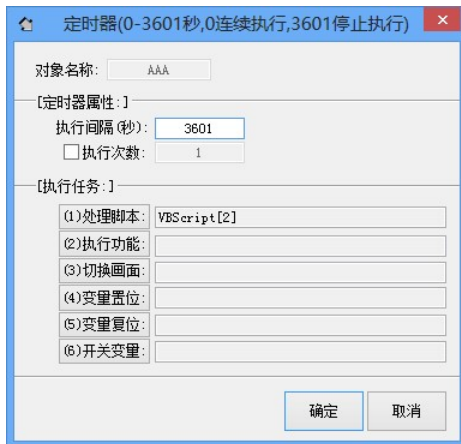
```
UserDB.ImportRecordsFromCsvFile
```

```
dbName, tableName, mainPath, subPath, fileName, fileStyle, rarPath, delFile, clrTable, endVariable
```

```
RunSys.SetDatabaseLink "连接1.udl", "UdlReadMode=2"
```

```
RunSys.SetVarValue VA, "%VA4", -1, 0
```

- [定时器]执行脚本触发数据库连接读取记录到内部变量:



```
RunSys.Var_Add_n VA, "%VA4", -1, 1
```

```
RunSys.RunDatabaseLink "连接 1.udl"
```

- [播放]按钮启动定时器秒计时:

```
TimerObj.SetTimer "AAA", 1
```

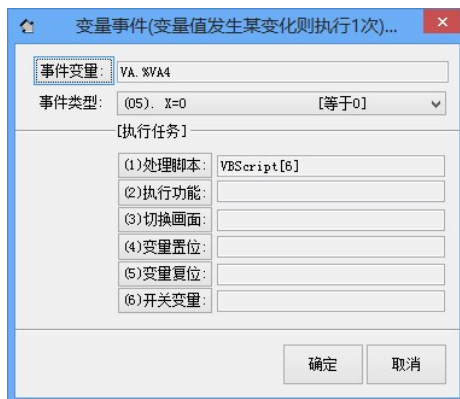
- [快进]按钮使定时器尽快执行:

```
TimerObj.SetTimer "AAA", 0
```

- [停止]按钮停止定时器:

```
TimerObj.SetTimer "AAA", 3601
```

- 数据库连接读记录到达最后, 复位记录号变量, 通过[变量事件]停止定时器:




```
TimerObj.SetTimer "AAA", 3601
```



### 9.13 画面回放(实时数据库)

- 实时数据库连接, 存储变量当前值到实时数据库;



实时数据库连接 - [连接1.rtl]

连接(F)

编辑(E)

窗口(W)

序号	标签名称 (Tag)	对应变量	类型	压缩	触发方式
1	TAG1	VA.XRAND	Real	×	时间
2	TAG2	VA.XI=X+1	Real	×	时间
3	TAG3	VA.XI=X-1	Real	×	时间

实时数据库参数

服务器地址: (local)

TCP端口号: 5878

连接超时 (s): 3

重连间隔 (s): 10

循环间隔 (ms): 100

标签包容量: 300

初始连接状态: 1 - 启动

连接服务器...

实时数据库标签

标签名称: TAG3

标签描述:

标签类型: 0 - 普通

标签标识: 18

数值类型: 3 - Real

对应变量: VA.XI=X-1

☐ 允许压缩
 

压缩误差: 0

☒ 时间触发
 

起始时间: 00:00:00
 

↑

↓

周期 (秒): 1

☐ 变化触发
 

变化值: 0
 

↑

↓

持续 (秒): 0

☐ 激活条件:

添加

修改

删除

- 制作回放测试画面:

2014-11-24

20:23:00

启动回放

停止回放



---

时间:

VT. %VT3

变量1:

VA. %VA1

变量2:

VA. %VA2

变量3:

VA. %VA3

定时器缺省禁止执行:

[illegible]

- [启动回放]按钮执行脚本:

```
' 设置开始时间,启动秒定时器
sDate=RunSys.GetVTtext("%VT1",-1)
sTime=RunSys.GetVTtext("%VT2",-1)
DatetimeObj.SetInitTimeEx sDate&" "&sTime
TimerObj.SetTimer "AAA",1
```

- [定时器]脚本:

```
' 回放时间前进1秒
DatetimeObj.MoveBack 0,0,0,1
' 得到回放时间
timeDesc=DatetimeObj.GetFormatTime("%Y-%m-%d %H:%M:%S")
RunSys.SetVTtext "%VT3",-1,timeDesc
' 连接实时数据库,如果已被连接,立即返回
n=FameHistoryObj.Connect("127.0.0.1",5678,"admin","admin",3)
If n>0 Then
    ' 从实时数据库获取历史数据,缓存到数组
    ReDim valueArray(3)
    n=FameHistoryObj.GetHistoricalValues("tag1|tag2|tag3|",timeDesc,valueArray)
    ' 数组中历史数据,写入内部回放变量,进行显示
    If n>0 Then
        n=RunSys.SetMultipleVarValue("va.%VA1|va.%va2|va.%va3",valueArray)
    Else' 读取实时数据库失败,停止回放
        FameHistoryObj.DisConnect
        TimerObj.SetTimer "AAA",3601
    End If
Else' 连接实时数据库失败,停止回放
    TimerObj.SetTimer "AAA",3601
End If
```

- [停止回放]按钮执行脚本:

```
' 停止定时器,断开实时数据库连接
TimerObj.SetTimer "AAA",3601
FameHistoryObj.DisConnect
```

- 运行效果:

2014-11-24	17:31:00	启动回放	停止回放
------------	----------	------	------

时间:	2014-11-24 17:31:04
变量1:	3935.67
变量2:	565.00
变量3:	565.00

### 9.14 根据报警状态制作故障诊断

□ 定义变量报警：

定义变量报警(支持30000条报警)...

编号	变量	类型	位置	名称	提示信息
1	DR_DR1	正跳变		DR1	正跳变报警
2	DR_DR2	正跳变		DR2	正跳变报警
3	DR_DR3	正跳变		DR3	正跳变报警
4	DR_DR4	正跳变		DR4	正跳变报警
5	DR_DR5	正跳变		DR5	正跳变报警
6	DR_DR6	正跳变		DR6	正跳变报警
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					
22					
23					
24					
25					

报警变量: DR\_DR1

报警类型[变量组成时定义]:

☐ HH - 超高报警 ☐ 1~>0 负跳变 ☐ 增幅报警  
☐ H - 高报警 ☒ 0~>1 正跳变 ☐ 陡增报警  
☐ L - 低报警 ☐ 0~-1 变位 ☐ 陡减报警  
☐ LL - 超低报警

报警定义:

级别: 2 - 报警  
位置:   
名称: DR1  
信息: 正跳变报警  
备注:

报警表现方式:

☒ 信息存储[报警系统允许被存储的情况下].  
☐ 执行功能/触发变量(产生报警+1, 确认或恢复-1):  
☐ 播放声音文件[支持多文件, 用' '或' \ '分隔]:  
☐ 循环播放, 直到被确认/恢复/次:  
☐ 播放条件:  
☐ 弹出对话框[标题/信息]: ☐ 仅提示而不确认  
☐ 发送手机短信.  
☐ 行打印机输出(不建议):  
报警延迟时间[秒]: 0  
☐ 报警条件:

保存

□ MyFile 目录下编辑故障分析文件, 每报警编号对应一组诊断信息:

```
故障诊断 - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

[Alarm_1]
Info = 上料皮带跳闸
Tip1 = 11、检查上料皮带是否卡阻
Tip2 = 12、检查电机三相绕组电阻是否一致
Tip3 = 13、检查电机三相绕组绝缘是否损坏
Tip4 = 14、检查电机三相绕组绝缘是否损坏
var = %VA2

[Alarm_2]
Info = 上料皮带跳闸
Tip1 = 21、检查上料皮带是否卡阻
Tip2 = 22、检查电机三相绕组电阻是否一致
Tip3 = 23、检查电机三相绕组绝缘是否损坏

[Alarm_3]
Info = 上料皮带跳闸
Tip1 = 31、检查上料皮带是否卡阻
Tip2 = 32、检查电机三相绕组电阻是否一致
Tip3 = 33、检查电机三相绕组绝缘是否损坏

[Alarm_4]
Info = 上料皮带跳闸
Tip1 = 41、检查上料皮带是否卡阻
Tip2 = 42、检查电机三相绕组电阻是否一致
Tip3 = 43、检查电机三相绕组绝缘是否损坏

[Alarm_5]
Info = 上料皮带跳闸
Tip1 = 51、检查上料皮带是否卡阻
Tip2 = 52、检查电机三相绕组电阻是否一致
Tip3 = 53、检查电机三相绕组绝缘是否损坏

[Alarm_6]
Info = 上料皮带跳闸
Tip1 = 61、检查上料皮带是否卡阻
Tip2 = 62、检查电机三相绕组电阻是否一致
Tip3 = 63、检查电机三相绕组绝缘是否损坏
```

□ 画面中建立报警状态组件：

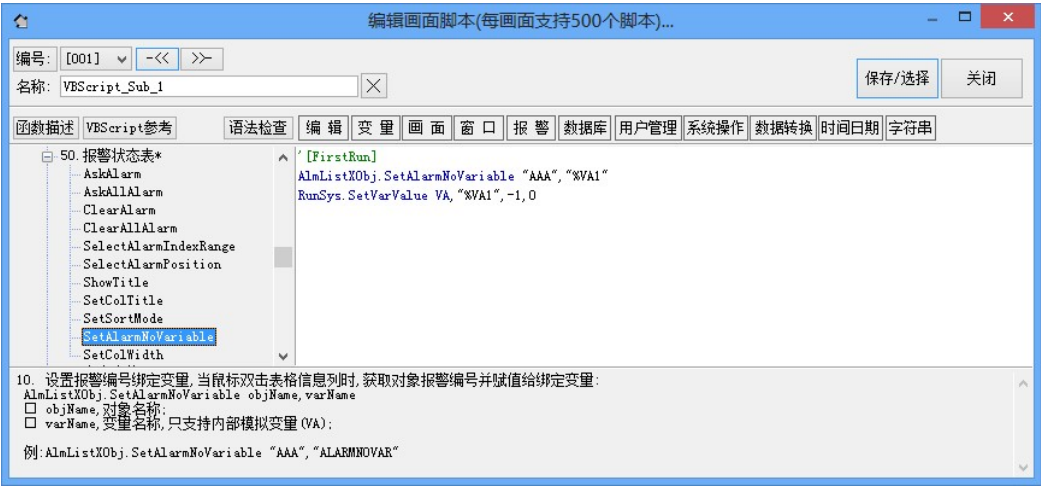
编号	状态	报警时间	确认/恢复时间	报警位置	报警名称	报警信息	报警值
						[鼠标双击, 移动鼠标到竖线, 可设置列	

□ 选择组件, 执行右键->增强属性->组件名称, 为组建命名：



□ 编写画面首次运行脚本, 把某 VA 变量绑定到报警状态表；

当双击表格的编号、信息、名称列时, 获取相应报警号赋值给绑定 VA 变量：



```
' [FirstRun]
AlmListXObj.SetAlarmNoVariable "AAA", "%VA1"
RunSys.SetVarValue VA, "%VA1", -1, 0
```

□ 根据绑定 VA 变量, 建立变量事件:



当 VA 变量发生任何变化出发脚本, 从诊断文件中获取报警信息, 提示对话框显示:

```
x=RunSys.GetVarValue(VA,"%VA1",-1)
If x>0 Then
    s=INIFile.GetStringText("故障诊断.txt","Alarm_"&CStr(x),"Info")
    s1=INIFile.GetStringText("故障诊断.txt","Alarm_"&CStr(x),"Tip1")
    s2=INIFile.GetStringText("故障诊断.txt","Alarm_"&CStr(x),"Tip2")
    s3=INIFile.GetStringText("故障诊断.txt","Alarm_"&CStr(x),"Tip3")
    ts=s&vbCrLf
    If Len(s1)>0 Then ts=ts&s1&vbCrLf
    If Len(s2)>0 Then ts=ts&s2&vbCrLf
    If Len(s3)>0 Then ts=ts&s3&vbCrLf
    n=MsgBox(ts,4096,"系统提示")
    If n=1 Then
        s4=INIFile.GetStringText("故障诊断.txt","Alarm_"&CStr(x),"var")
        If Len(s4)>0 Then RunSys.SetVarValue VA,s4,-1,1
    End If
    RunSys.SetVarValue VA,"%VA1",-1,0
End If
```

□ 运行效果:

故障诊断

编号	状态	报警时间	确认/恢复时间	位置	名称	信息	报警值
1	×	2014-06-06 09:42:26				正跳变报警	1
2	×	2014-06-06 09:42:26				正跳变报警	1
3	×	2014-06-06 09:42:27				正跳变报警	1
4	×	2014-06-06 09:42:27				正跳变报警	1
5	×	2014-06-06 09:42:28				正跳变报警	1
6	×	2014-06-06 09:42:29				正跳变报警	1

系统提示

上料皮带跳闸  
31、检查上料皮带是否卡阻  
32、检查电机三相绕组电阻是否一致  
33、检查电机三相绕组绝缘是否损坏

确定

9.15 实现设备维护提醒画面

- 假设有两个设备,需统计其工作时间及启停次数;
- 建立相关变量,使用 CA 计算功能进行统计(如果有 PLC 控制器,通过在 PLC 中统计会更准确):

	设备 1	设备 2
启停状态	DR. 设备 1_启停状态	DR. 设备 2_启停状态
计时变量	VA. @设备 1_运行时间	VA. @设备 2_运行时间
	CA. 设备 1_计时	CA. 设备 2_计时
计数变量	VA. @设备 1_启停次数	VA. @设备 2_启停次数
	CA. 设备 1_计数	CA. 设备 @_计数

- CA 功能内容:



- 在 PictureFile 目录下,建立文本文件,例如“设备维护.txt”:

```
*****  
;基本参数:  
;fileType=1,      此文件用于配置设备维护表  
;fontName         字体名称  
;fontSize         字体大小  
;backColor        背景颜色, RGB(r, g, b)  
;textColor        文本颜色, RGB(r, g, b)  
;fixRows          固定行数量(1)  
;fixCols          固定列数量(1-7)  
;fixBackColor     固定行列背景颜色, RGB(r, g, b)  
;fixTextColor     固定行列文本颜色, RGB(r, g, b)  
;lineColor        网格颜色, RGB(r, g, b), 特殊颜色 RGB(254, 254, 254) 时不显示网格
```

```

;rowHeight      行高(300-1000)
;colWidth       列宽(1000-15000), 每字符大概 150 点
;colTitle       7 列标题(编号, 名称, 设定值, 实际值, 单位, 状态, 复位时间)
;colAlignment   列对齐方式(0=中, 1=左, 2=右)
;statueTextDesc 状态文本描述, 四级别(正常, 提示, 警告, 报警)
;alarmBackColor 报警行背景颜色, 三报警(提示|警告|报警)=(rgb) (rgb) (rgb)
;alarmTextColor 报警行文本颜色, 三报警(提示|警告|报警)=(rgb) (rgb) (rgb)
;deviceRows     设备行数量(1-3000)
;updateTime     刷新秒间隔(1-600)

;*****
[Basic Parameters]
fileType        = 1
fontName        = 宋体
fontSize        = 9
backColor       = (255, 255, 255)
textColor       = (0, 0, 0)
fixRows         = 1
fixCols         = 0
fixBackColor    = (191, 219, 252)
fixTextColor    = (0, 0, 0)
lineColor       = (192, 192, 192)
rowHeight       = 380
colWidth        = 800, 3200, 1200, 1200, 1000, 1000, 2200
colTitle        = 编号, 名称, 设定值, 实际值, 单位, 状态, 复位时间
colAlignment    = 0, 1, 2, 2, 0, 0, 0
tatusTextDesc   = 正常, 提示, 警告, 报警
alarmBackColor  = (255, 200, 200), (255, 150, 0), (255, 50, 0)
alarmTextColor  = (0, 0, 0), (0, 0, 0), (255, 255, 255)
deviceRows      = 4
updateTime      = 1

;*****
;行号内容:
;deviceNo       设备编号, 缺省为行号
;deviceName     设备名称或描述
;setValue       预设维护值
;varName        设备状态对应变量(XX.YYYYYY)
;kValue         系数值:/3600, *10, /60, *1, +100, -100, 缺省为*1

```

```

;valueFormat    显示格式,数值格式(0-6):%.nf, 时间格式(10):%H:%M:%S,
                  时间格式(11):%d.%H:%M:%S, 缺省(0)

;valueUnit      数值单位

;resetTime      复位时间,自动判断产生

;alarmStatus    当前报警状态,自动判断产生,0=正常,1=提醒,2=警告,3=报警;

*****

[Row_1]
deviceNo       = 10001
deviceName     = 设备 1 运行时间
setValue       = 3000
varName        = VA.@设备 1_运行时间
kValue         = /3600
valueFormat    = 4
valueUnit      = Hour
statusRange    = 2000,2500,2800
resetTime      =
alarmStatus    = 0

[Row_2]
deviceNo       = 10002
deviceName     = 设备 1 启停次数
setValue       = 9000
varName        = VA.@设备 1_启停次数
kValue         =
valueFormat    = 0
valueUnit      = Times
statusRange    = 7000,8000,8500
resetTime      = 2014-10-31 21:33:30
alarmStatus    = 0

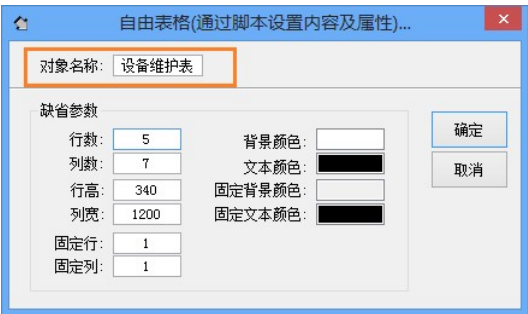
[Row_3]
deviceNo       = 10003
deviceName     = 设备 2 运行时间
setValue       = 3000
varName        = VA.@设备 2_运行时间
kValue         = /60
valueFormat    = 2
valueUnit      = Minute
statusRange    = 2000,2500,2800
resetTime      =
alarmStatus    = 0

```



[Row\_4]  
deviceNo = 10004  
deviceName = 设备 2 启停次数  
setValue = 9000  
varName = VA.@设备 2\_启停次数  
kValue =  
valueFormat = 0  
valueUint = Times  
statusRange = 7000,8000,8500  
resetTime =  
alarmStatus = 0

□ 画面中添加自由表格控件, 并命名, 其他参数缺省:



□ 画面中使用初始脚本, 通过配置文件定制自由表格:



□ 测试画面如下:

设备1\_启动

设备2\_启动

编号	名称	设定值	实际值	单位	状态	复位时间
10001	设备1运行时间	3000	0.2892	Hour	正常	
10002	设备1启停次数	9000	8500	Times	报警	2014-10-31 21:33:30
10003	设备2运行时间	3000	2.42	Minute	正常	
10004	设备2启停次数	9000	7	Times	正常	2014-10-31 21:33:30

- 如果计时或计数变量为 VA/AR 变量, 可实现复位功能:

设备1_停止		设备2_停止		单个复位		全部复位
编号	名称	设定值	实际值	单位	状态	复位时间
10001	设备1运行时间	3000	0.3031	Hour	正常	
10002	设备1启停次数	9000	8523	Times	报警	2014-11-01 17:16:06
10003	设备2运行时间	3000	3.23	Minute	正常	
10004	设备2启停次数	9000	8636	Times	报警	2014-11-01 17:16:06

单个复位脚本:

```
' 获取当前选择的自由表格行号
row=GridListXObj.GetSelRow("设备维护表")
' 根据行号获取相应设备是否处于报警状态
path=RunSys.GetVTtext ("%SYSTEMDIR", -1)
alarmStatus=INIFile.GetStringText (path&"\picturefile\设备维护.txt", "Row_"&CStr(row), "alarmStatus")
If RunSys.ATI (alarmStatus)>1 Then
    ' 根据行号获取相应变量名称
    varName=INIFile.GetStringText (path&"\picturefile\设备维护.txt", "Row_"&CStr(row), "varName")
    ' 复位变量值
    RunSys.SetVarValue 0, varName, -1, 0
End If
```

全部复位脚本:

```
' 获取设备总数
path=RunSys.GetVTtext ("%SYSTEMDIR", -1)
s=INIFile.GetStringText (path&"\picturefile\设备维护.txt", "Basic Parameters", "deviceRows")
rows=RunSys.ATI (s)
' 检查全部报警状态并复位
For n=1 To rows
    alarmStatus=INIFile.GetStringText (path&"\picturefile\设备维护.txt", "Row_"&CStr(n), "alarmStatus")
    If RunSys.ATI (alarmStatus)>1 Then
        ' 根据行号获取相应变量名称
        varName=INIFile.GetStringText (path&"\picturefile\设备维护.txt", "Row_"&CStr(n), "varName")
        ' 复位变量值
        RunSys.SetVarValue 0, varName, -1, 0
    End If
Next
```

## 9.16 拷贝画面内容到 Word 文档

' 复制画面位置 (0, 0, 800, 600) 内容到粘贴版

Picture.SaveasBmp 0, 0, 800, 600, 128, 128, 128, "", "pasteboard.bmp"

' 启动Word

Set WordObj = CreateObject("Word.Application")

' 隐藏Word窗口

WordObj.Visible = False

' 添加空Word文档

Set wdBook=WordObj.Documents.Add

' 把粘贴板内容复制到Word文档

WordObj.Selection.Paste

' 另存文档

fileName=CurrentTime.GetFormatTime("%Y%m%d\_%H.doc")

wdBook.SaveAs "C:\fameView\temp\"&fileName

' 关闭文档并退出Word

wdBook.Close False

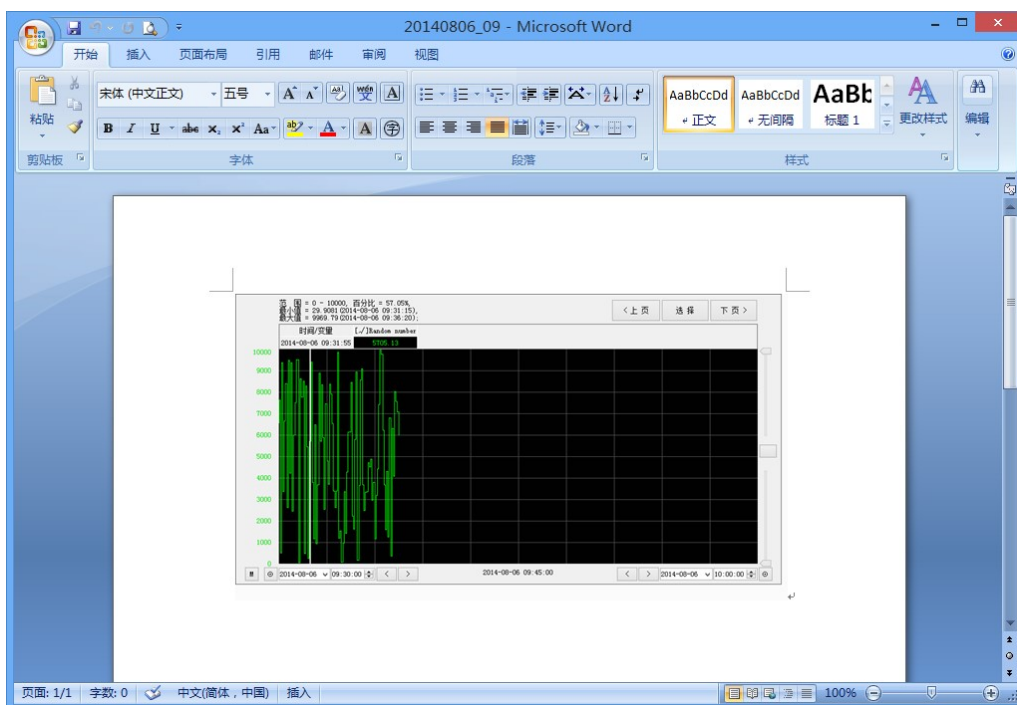
WordObj.Quit

' 销毁Word对象

Set wdBook=Nothing

Set WordObj=Nothing

Word文件内容:

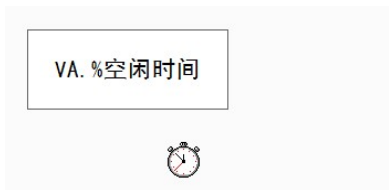


### 9.17 画面某段时间无操作切换到其他画面

- 提供内部系统模拟变量(VA. 空闲时间), 内容为无鼠标及键盘操作的秒计时;



- 画面添加定时器:



- 定时器每隔 3 秒执行 1 次脚本, 脚本中检查 VA. %空闲时间>60 秒时, 切换到其他画面:



```
x=RunSys.GetVarValue(VA, "%空闲时间", -1)
```

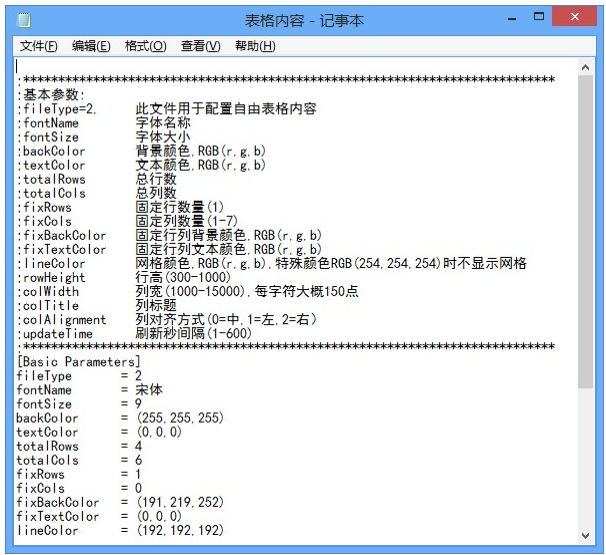
```
If x>60 Then
```

```
    Picture.SwitchToPicture "other.drw", 1
```

```
End If
```

### 9.18 通过文本文件配置自由表格

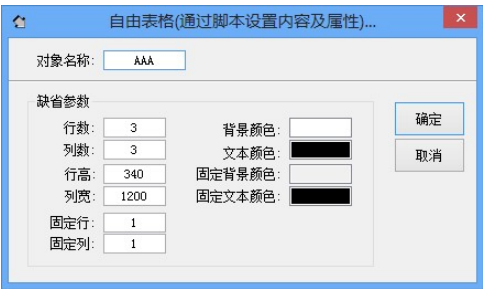
□ PictureFile 目录下, 建立文本文件, 例如“表格内容.txt”:



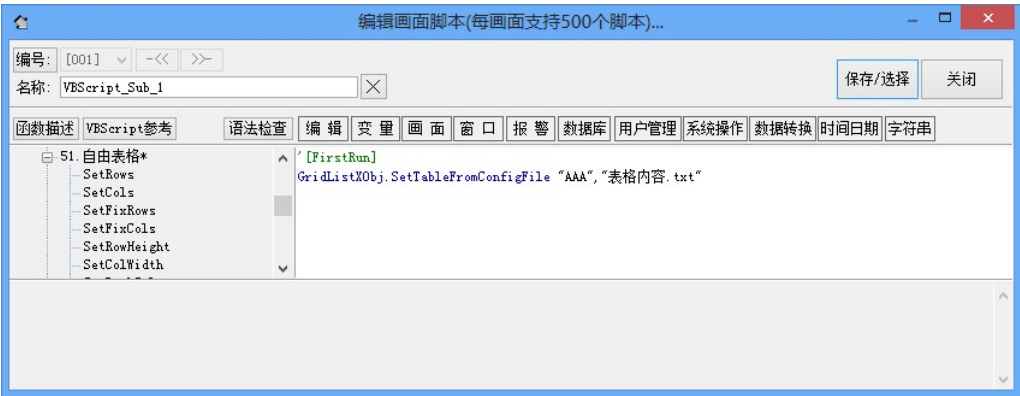
```
*****
;基本参数:
;fileType=2,      此文件用于配置自由表格内容
;fontName         字体名称
;fontSize         字体大小
;backColor        背景颜色, RGB(r, g, b)
;textColor        文本颜色, RGB(r, g, b)
;totalRows        总行数
;totalCols        总列数
;fixRows          固定行数量(1)
;fixCols          固定列数量(1-7)
;fixBackColor     固定行列背景颜色, RGB(r, g, b)
;fixTextColor     固定行列文本颜色, RGB(r, g, b)
;lineColor        网格颜色, RGB(r, g, b), 特殊颜色 RGB(254, 254, 254) 时不显示网格
;rowHeight        行高(300-1000)
;colWidth         列宽(1000-15000), 每字符大概150点
;colTitle         列标题
;colAlignment     列对齐方式(0=中, 1=左, 2=右)
;updateTime       刷新秒间隔(1-600)
*****
[Basic Parameters]
fileType          = 2
fontName          = 宋体
fontSize          = 9
backColor         = (255, 255, 255)
textColor         = (0, 0, 0)
totalRows         = 4
totalCols         = 6
fixRows           = 1
fixCols           = 0
fixBackColor      = (191, 219, 252)
fixTextColor      = (0, 0, 0)
lineColor         = (192, 192, 192)
;*****
;基本参数:
;fileType=2,      此文件用于配置自由表格内容
;fontName         字体名称
;fontSize         字体大小
;backColor        背景颜色, RGB(r, g, b)
;textColor        文本颜色, RGB(r, g, b)
;totalRows        总行数
;totalCols        总列数
;fixRows          固定行数量(1)
;fixCols          固定列数量(1-7)
;fixBackColor     固定行列背景颜色, RGB(r, g, b)
;fixTextColor     固定行列文本颜色, RGB(r, g, b)
;lineColor        网格颜色, RGB(r, g, b), 特殊颜色 RGB(254, 254, 254) 时不显示网格
;rowHeight        行高(300-1000)
;colWidth         列宽(1000-15000), 每字符大概 150 点
;colTitle         列标题
;colAlignment     列对齐方式(0=中, 1=左, 2=右)
;updateTime       刷新秒间隔(1-600)
[Basic Parameters]
fileType          = 2
fontName          = 宋体
fontSize          = 9
```

```
backColor      = (255,255,255)
textColor      = (0,0,0)
totalRows      = 4
totalCols      = 6
fixRows        = 1
fixCols        = 0
fixBackColor   = (191,219,252)
fixTextColor   = (0,0,0)
lineColor      = (192,192,192)
rowHeight      = 380
colWidth       = 800,1200,1200,1000,1000,1000
colTitle       = 编号,标题1,标题2,标题3,标题4,标题5
colAlignment   = 0,1,1,2,2,2
updateTime     = 2
;*****
;表格内容:行中可包含文本或变量,变量后附加小数位数(0-4),(100-104)表示变量值0时空格;
[Table Context]
Row_1 = 1|文本11|文本12|VA.%RAND(2)|VA.%RAND(1)|VA.%RAND(0)
Row_2 = 2|文本21|文本22|VA.%RAND(2)|VA.%RAND(1)|VA.%RAND(0)
Row_3 = 3|文本31|文本32|VA.%RAND(2)|VA.%RAND(1)|VA.%RAND(0)
```

□ 画面中添加自由表格控件,并命名,其他参数缺省:

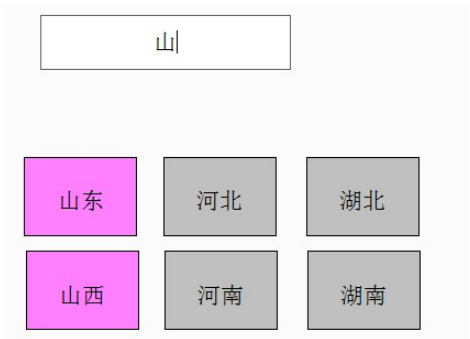


□ 画面中使用初始脚本,通过配置文件定制自由表格:



### 9.19 模糊查找画面对象

- 输入对象名称时, 符合条件的对象变色:



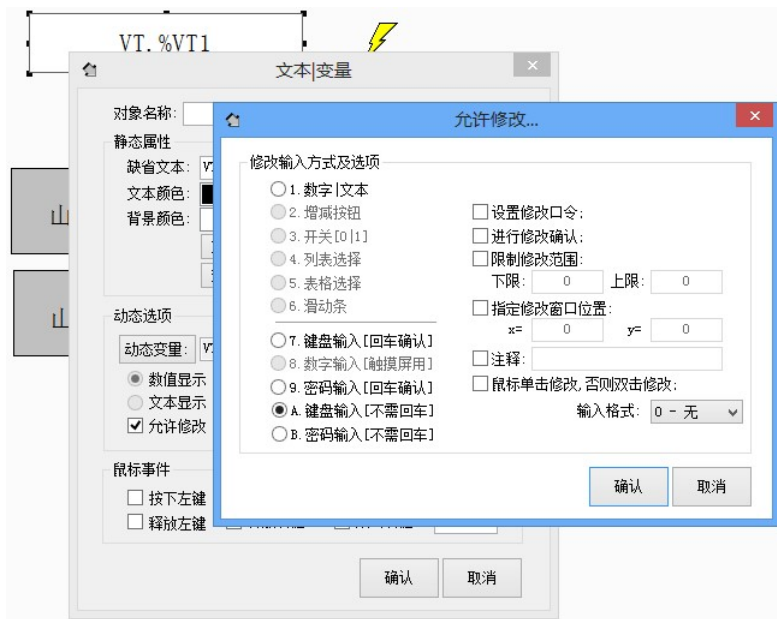
- 制作以下画面:



- 每个画面对象由矩形和标签组成, 矩形对象名称与标签内容相同:



- VT 变量允许修改, 并采用键盘输入方式:



- 根据文本变量做变量事件:



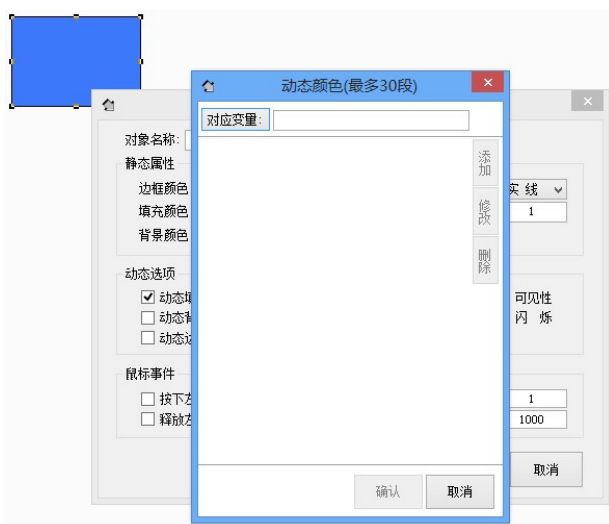
- 变量发生任何变化, 执行脚本:

```
s=RunSys.GetVTtext ("%VT1",-1)
If s="" Then
    ShapeObj.SetFillColor "*",192,192,192
Else
    ShapeObj.SetFillColor s&"$",192,192,192
    ShapeObj.SetFillColor s&"*",255,128,255
End If
```

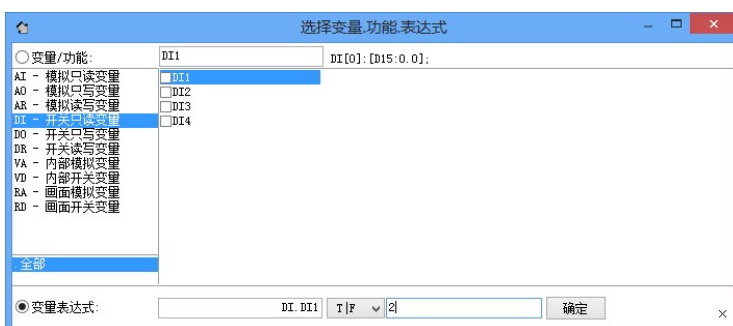


## 9.20 变量无效动态颜色指示

- 模拟变量(AI/AO/AR)和开关变量(DI/DO/DR),其数值无法表达通讯中断状态,需特殊方法实现;
- 例如要通过变量 DI、DI1,使矩形填充色三种颜色指示(0=红色,1=绿色,无效=灰色):

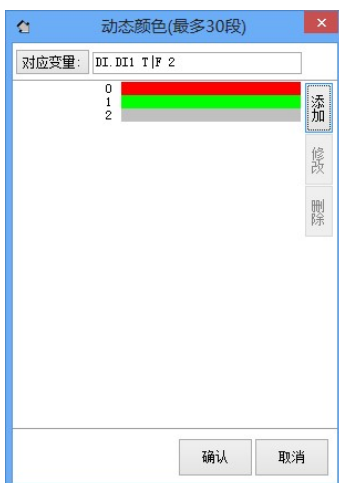


- 执行[对应变量]按钮,选择[变量表达式]并输入:



当变量 DI.DI1 有效(Ture)时, 表达式值取 0/1, 当变量 DI.DI1 无效效(False)时, 表达式值取 2;

- 根据表达式设定动态颜色:



9.21 动态修改树形框 XML 文件

配方

A

B

C

D

E

F

other

增加

修改

删除

配方编号:

配方名称:

粉料

1

0.00

0.00

0.00

0.00

1

0.00

0.00

0.00

0.00

1

0.00

0.00

0.00

0.00

1

0.00

0.00

0.00

0.00

1

0.00

0.00

0.00

0.00

油料

1

0.00

0.00

0.00

1

0.00

0.00

0.00

1

0.00

0.00

0.00

胶料

0.00

0.00

0.00

0.00

0.00

工艺

0 - 无

0 - 无

0.00

0.00

0.00

0.00

0.00

0 - 无

0 - 无

0.00

0.00

0.00

0.00

0.00

0 - 无

0 - 无

0.00

0.00

0.00

0.00

0.00

0 - 无

0 - 无

0.00

0.00

0.00

0.00

0.00

0 - 无

0 - 无

0.00

0.00

0.00

0.00

0.00

0 - 无

0 - 无

0.00

0.00

0.00

0.00

0.00

XML 文件格式:

```
- <FameView_TreeCtrl_XmlFile>
- <_配方 id="0" fun="">
  <_A id="0" fun=""/>
  <_B id="0" fun=""/>
  <_C id="0" fun=""/>
  - <_D id="0" fun="">
    <_D0001-配方1 id="1" fun=""/>
  </_D>
  <_E id="0" fun=""/>
  - <_F id="0" fun="">
    <_F0002-fffff id="1" fun=""/>
  </_F>
  <_other id="0" fun=""/>
</_配方>
</FameView_TreeCtrl_XmlFile>
```

,

' 增加内容:

```
Sub AddRecipeToTree
' 打开 TreeFile.xml
path=RunSys.GetInstallPath()
Set doc = CreateObject("Microsoft.XMLDOM")
doc.async = False
doc.Load path&"\picturefile\treefile.xml"
' 找到根节点
Set root = doc.SelectSingleNode("FameView_TreeCtrl_XmlFile")
If root Is Nothing Then
  RunSys.ShowMsgWnd "添加配方错误:treefile.xml 格式错误(1)!", 3
Set doc=Nothing:Exit Sub
```

```

End If
' 找到次根节点
Set root1 = root.SelectSingleNode("_配方")
If root1 Is Nothing Then
    RunSys.ShowMsgWnd "添加配方错误:treefile.xml 格式错误(2)!", 3
    Set doc=Nothing:Exit Sub
End If
' 找到配方号第 1 字节节点
Set node = root1.SelectSingleNode("_"&Left(recipeNo, 1))
If node Is Nothing Then
    Set node = root1.SelectSingleNode("other")
    If node Is Nothing Then
        RunSys.ShowMsgWnd "添加配方错误:treefile.xml 格式错误(3)!", 3
        Set doc=Nothing:Exit Sub
    End If
End If
' 添加配方节点
Set node1=doc.CreateElement("_"&recipeNo&"-"&recipeName)
node.AppendChild node1
node1.setAttribute "id", "1"
node1.setAttribute "fun", ""
' 保存文件
doc.Save path&"\picturefile\treefile.xml"
Set doc=Nothing
' 重新装载
TreeObj.RefillTree "AAA", path&"\picturefile\treefile.xml"
End Sub
' -----
' 删除内容:
Sub DeleteRecipeFromTree
    ' 打开 TreeFile.xml
    path=RunSys.GetInstallPath()
    Set doc = CreateObject("Microsoft.XMLDOM")
    doc.async = False
    doc.Load path&"\picturefile\treefile.xml"
    ' 找到根节点
    Set root = doc.SelectSingleNode("FameView_TreeCtrl_XmlFile")
    If root Is Nothing Then
        Set doc=Nothing:Exit Sub
    End If

```

```
End If
' 找到次根节点
Set root1 = root.SelectSingleNode("_配方")
If root1 Is Nothing Then
    Set doc=Nothing:Exit Sub
End If
' 找到配方号第 1 字节节点
Set node = root1.SelectSingleNode("_"&Left(recipeNo,1))
If node Is Nothing Then
    Set node = root1.SelectSingleNode("other")
    If node Is Nothing Then
        Set doc=Nothing:Exit Sub
    End If
End If
' 找到配方节点
Set node1 = node.SelectSingleNode("_"&recipeNo&"-"&recipeName)
If node1 Is Nothing Then
    Set doc=Nothing:Exit Sub
End If
' 删除配方节点
node.RemoveChild node1
' 保存文件
doc.Save path&"\picturefile\treefile.xml"
Set doc=Nothing
' 重新装载
TreeObj.RefillTree "AAA",path&"\picturefile\treefile.xml"
End Sub
```

9.22 获取画面键盘输入内容

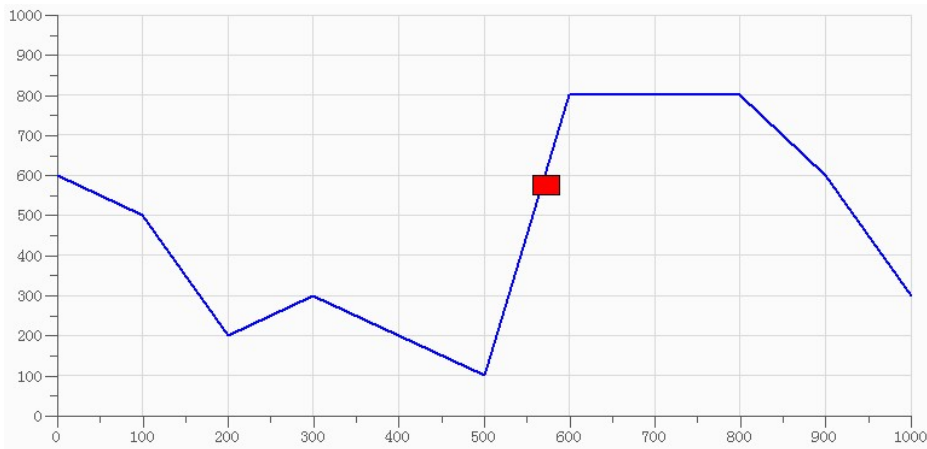
- ❑ 在画面中增加特殊画面文本变量(KeyboardInputText):

定义画面变量[1-1500]...			
编号	RA - 模拟变量	RD - 开关变量	RT - 文本变量
1			OBJNAME
2			CurVarMObjName
3			KeyboardInputText
4			
5			
-			

- ❑ 画面中键盘输入内容能够缓存在此画面文本变量中;
- ❑ Esc 键清除文本内容;
- ❑ 可通过脚本检查并控制文本内容;

9.23 设备沿运行轨迹移动

□ 需求: 预先设定某设备运行轨迹, 然后使设备根据位移计算出路径, 实现移动, 如下图;



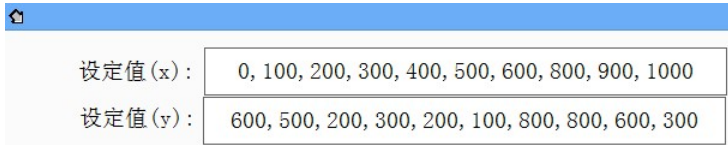
□ 制作两个具有记忆功能 VT 变量;



□ VT 变量在画面中作为轨迹设定值:



通过输入字符串分隔值进行设定值:



例子测试, 可使用更人性化的输入方式:

□ 画面添加温控曲线：



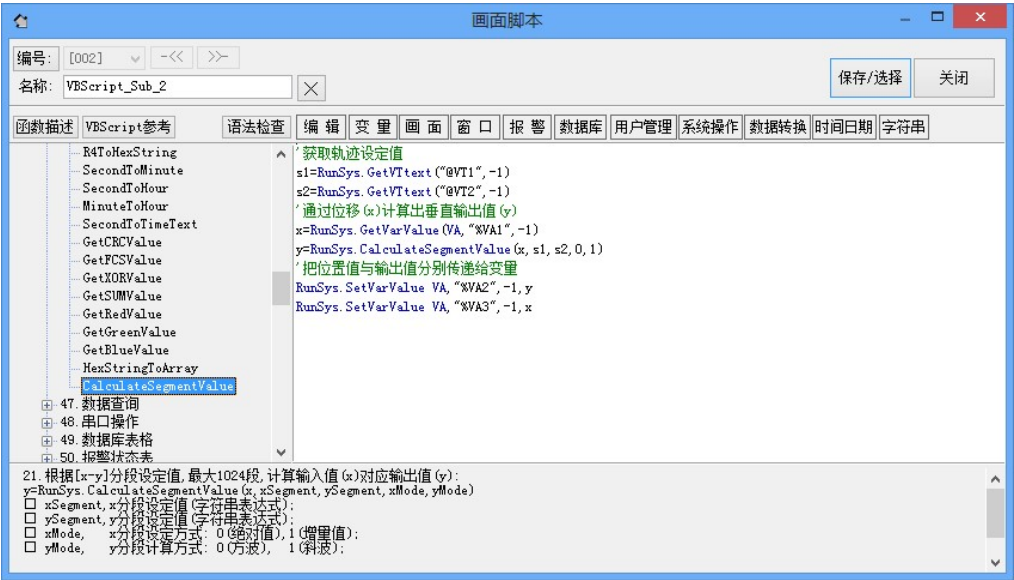
□ 通过[设定轨迹]按钮执行脚本, 使温控曲线与设定值关联：




□ 假设位移值变量(VA.%VA1), 则位移值变量变化时, 通过变量事件执行脚本：



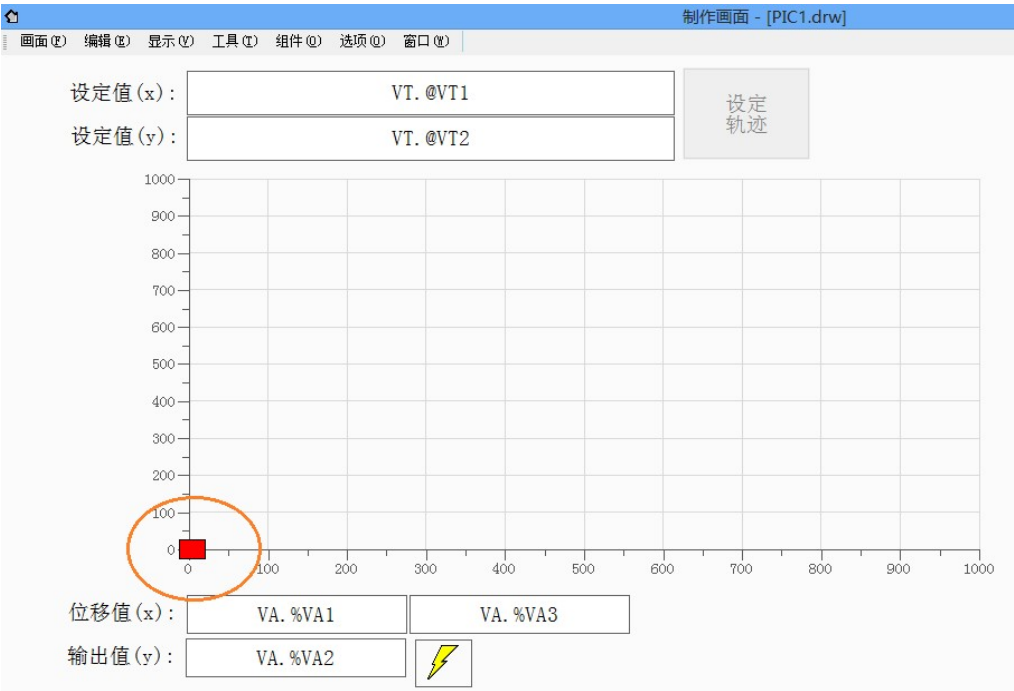
脚本获取轨迹位移变量值 (VA. %VA3) 和垂直输出值 (VA. %VA2) :



测试画面中输入及显示:

位移值(x):	VA. %VA1	VA. %VA3
输出值(y):	VA. %VA2	

□ 温控曲线上添加任意图形, 表示设备, 如矩形:



□ 设置图形水平与垂直移动属性：

矩形

对象名称: NONE

静态属性

边框颜色:

填充格式: 实 心

边框类型: 实 线

填充颜色:

背景格式: 实 心

边框宽度: 1

背景颜色:

动态选项

☐ 动态填充色

☐ 水平填充

☒ 水平移动

☐ 伸 缩

☐ 可见性

☐ 动态背景色

☐ 垂直填充

☒ 垂直移动

☐ 旋 转

☐ 闪 烁

☐ 动态边框色

鼠标事件

☐ 按下左键

☐ 按下右键

☐ 双击左键

扫描级别: 1

☐ 释放左键

☐ 释放右键

☐ 双击右键

释放延时: 1000

确定

取消

水平移动...

对应变数: VA. %VA3

移动偏移

变量低限: 0

对应像素偏移量: 0

变量高限: 1000

[-9999 ~ 9999] 710

确认

取消

垂直移动...

对应变数: VA. %VA2

移动偏移

变量低限: 0

对应像素偏移量: 0

变量高限: 1000

[-9999 ~ 9999] -335

确认

取消

变量高低限参考温控曲线位移和垂直量程, 像素偏移量参考温控曲线水平宽度与垂直高度像素;

□ 执行效果(输入设定值, 设定轨迹, 手动改变位移值):

设定值 (x): 0, 100, 200, 300, 400, 500, 600, 800, 900, 1000

设定值 (y): 600, 500, 200, 300, 200, 100, 800, 800, 600, 300

设定轨迹

位移值 (x): 586.00

586.00

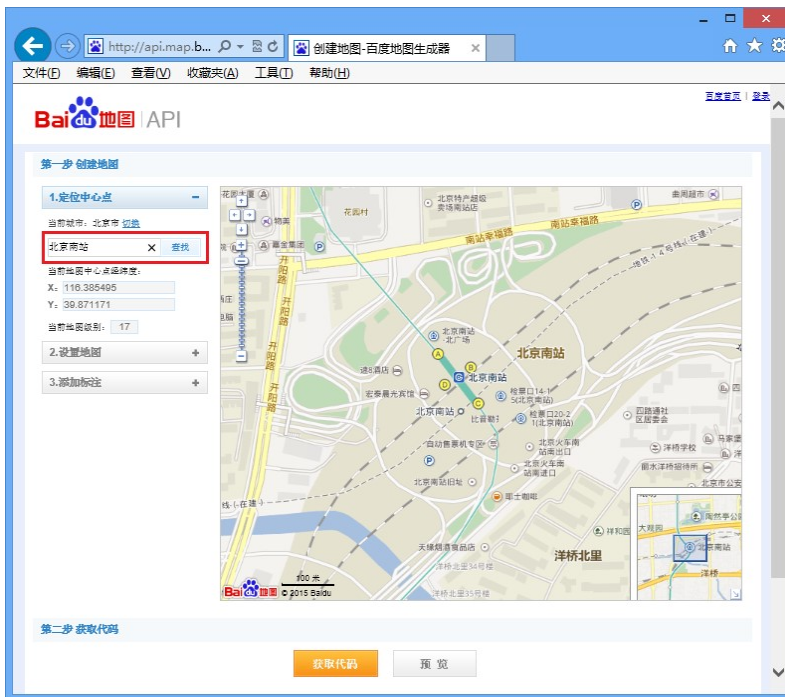
输出值 (y): 702.00

9-40

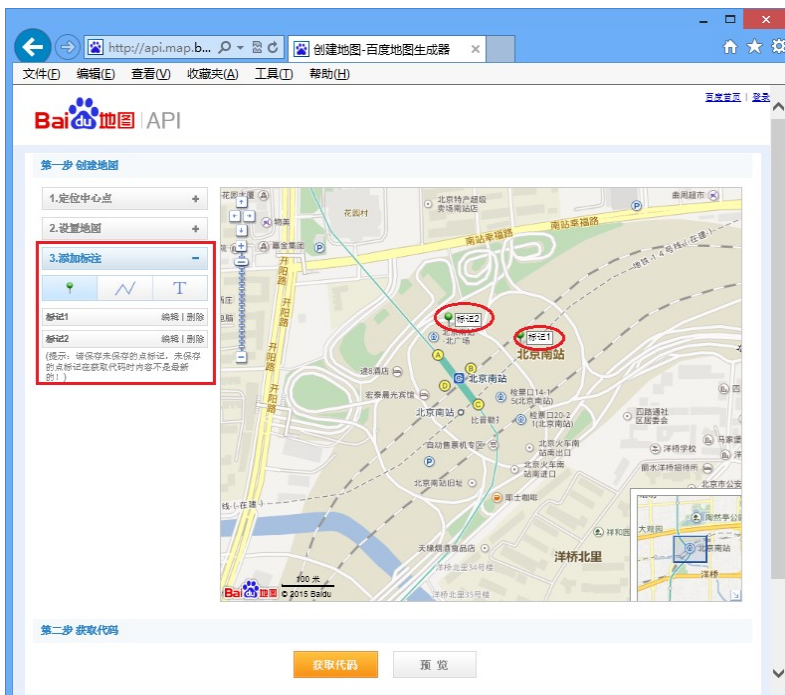


## 9.24 嵌入百度地图

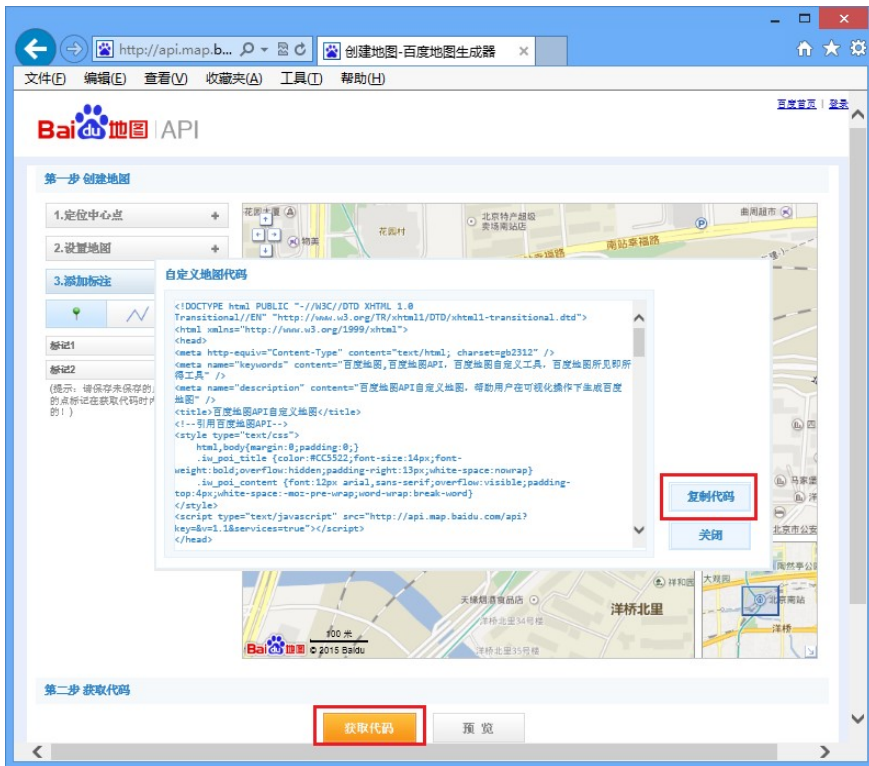
- 浏览器输入地址 (<http://api.map.baidu.com/lbsapi/creatmap/>) 并打开;
- 设置地图中心点位置, 如北京南站:



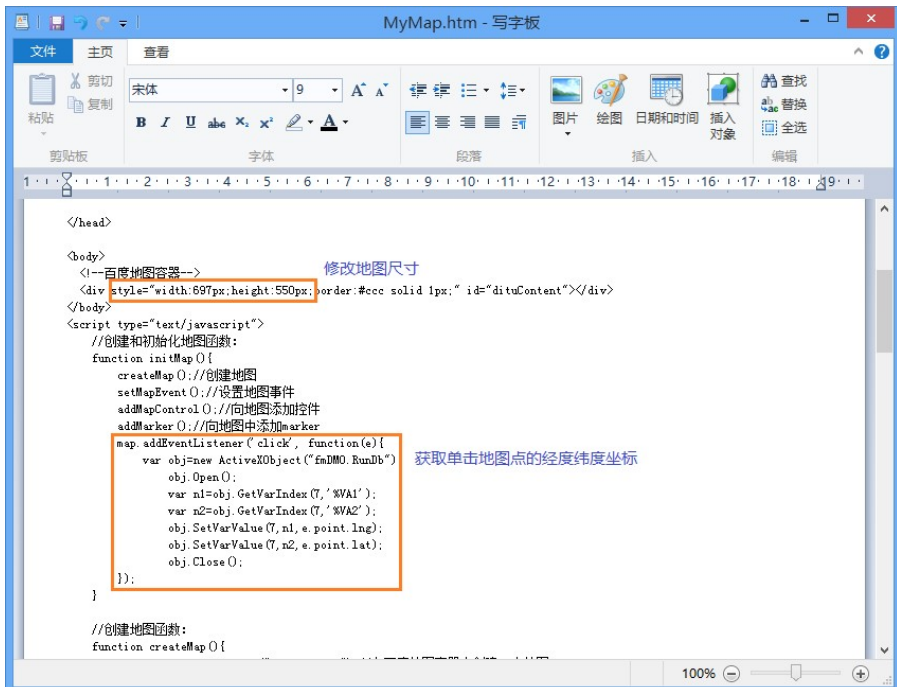
- 设置地图, 默认使用缺省, 添加标注:



- 执行[获取代码]按钮：



- 执行[复制代码]按钮；
- 打开写字板，粘贴复制内容，修改代码，设置地图尺寸并获取单击经纬度坐标到组态变量：



事件监听代码:

```
map.addEventListener('click', function(e) {
    var obj=new ActiveXObject("fmdm0.RunDb")
    obj.Open();
    var n1=obj.GetVarIndex(7,'%VA1');
    var n2=obj.GetVarIndex(7,'%VA2');
    obj.SetVarValue(7,n1,e.point.lng);
    obj.SetVarValue(7,n2,e.point.lat);
    obj.Close();
});
```

- 写字板内容保存到组态 PictureFile 目录下, 文件名例如 myMap.htm;
- 画面制作, 通过组件箱. 画面控件, 建立浏览器控件:



- 地图在线, 测试效果:



- 根据获取经纬度, 通过变量事件处理;