**脚本优先级（基于广州已使用但还未实现）**

**前言：所有脚本函数参数的obj必须带上obj所在的svm图名称，否则识别不出来，例如：”专业导航栏.1\_0\_文本\_12”。从22开始为广州还未使用的。**

##### 获取组代码(车站名缩写)char\* getgroupcode ()

功能：返回脚本所在屏的组代码

##### 获取图元的符号状态void getsymbolstatus (char \*obj)

功能：获取图元当前状态。

##### 获取图层显示状态int getlayervisible (char \*name, int index)

参数说明：

name：画面名（可省略）

index：图层序号（从1开始）

功能：取出名称等于name的svm图第index图层的显示状态，

0---未显示 1---已显示

备注：name为窗口打开的不包括目录和扩展名的SVM文件名，若省略画面名，系统默认为当前画面窗口。

##### 获取特定序号的界面全局整型点值init getvdataint (int no)

返回值：获取1-2047中no序号的界面全局虚拟点值。

##### 设置特定序号的界面全局整型点值void setvdataint (int num,int val)

参数说明：

num：1-2047的数字点号

val：整型值

功能：设置1-2047序号的界面全局虚拟点值

##### 设置特定序号的界面全局点字串void setvdatastr (int num,char\* val)

**参数说明：**

num：1-2047的数字点号

val：0-255字符的字串

**返回值：**设置1-2047序号的界面全局虚拟点字串

##### 获取特定序号的界面全局点字串void getvdatastr (int num)

**参数说明：**

num：1-2047的数字点号

**返回值：**获取1-2047序号的界面全局虚拟点字串

4个函数的作用：ehmi起来后会有两个全局的缓存区，int缓存区内1-2047个点绑定着int值，str缓存区内1-2047个点绑定者字符串。可以根据点号将值或者字符串存到缓存区，用的时候再根据点号取出值或者字符串。

##### 激活图元void enableobj (char \*obj, int enable)

功能：根据Obj（图元名称）设置图元是否有效，也就是能不能点击， enable：0—无效 1—有效

##### 获取图元的着色方案int getmetacolor (char \*obj)

功能：根据obj（图元名称，如0\_1\_文本\_813）得到图元着色标志，0-未着色，说明此时的颜色就是它本来的颜色；1-着色，说明此时的颜色经过了改变。

##### 设置图元的着色方案void setmetacolor (char \*obj, int r, int g, int b, int flag)

功能：根据obj（图元名称，如0\_1\_文本\_813）改变图元的背景色（r，g，b），flag=0则不改变，1才改变。

##### 获取图元显示状态int getobjvisible (char \*obj)

功能：返回显示标志，0---未显示 1---已显示。

##### 设置图元显示状态void setobjvisible (char \*obj, int visible)

功能：visible：图元是否显示，0---不显示1---显示。

##### 获取数字量点实时库域值int getdval(char \*PointCode)

功能：根据PointCode(点代码)取实时库di点表“Value”字段的值(一般，0-分闸，1-合闸，2-未知状态)。

##### 设置数字量点实时库域值void setdval(char \*PointCode, int value)

功能：根据PointCode(点代码)设置实时库di点表“Value”字段的值(一般，0-分闸，1-合闸，2-未知状态)。

##### 设置数字量点实时库域值void setdval0(char \*PointCode, char \*DESC, int value)

功能：先弹出二次确认框，框内文字为DESC，确认之后才根据PointCode(点代码)设置实时库di点表“Value”字段的值(一般，0-分闸，1-合闸，2-未知状态)；取消则退出函数。

##### 获取模拟量点实时库域值float getaval(char \*PointCode)

功能：根据PointCode(点代码)取实时库ai点表“Value”字段的值。

##### 设置模拟量点实时库域值void setaval(char \*PointCode, float value)

功能：根据PointCode(点代码)设置实时库ai点表“Value”字段的值。

##### 设置模拟量点实时库域值void setaval0(char \*PointCode, char \*DESC, int value)

功能：先弹出二次确认框，框内文字为DESC，确认之后才根据PointCode(点代码)设置实时库ai点表“Value”字段的值；取消则退出函数。

##### 获取两个值按位与的结果void getavaland (char \* AIcode,int val)

参数说明：

AIcode：AI点代码

Val：数值

返回值：返回AIcode和val的16位按位与后的值，用来判定16位为1的位

功能：根据AIcode得到ai点表“Value”字段的值，将得到的值跟Val做按位与操作。

##### 获取两个值按位或的结果void getavalor (char \* AIcode,int val)

参数说明：

AIcode：AI点代码

Val：数值

返回值：返回AIcode和val的16位按位或后的值，用来判定16位为0的位

功能：根据AIcode得到ai点表“Value”字段的值，将得到的值跟Val做按位或操作。

##### 打开图形窗口void openwnd (char \*name, int x1, int y1, int x2, int y2, int n)

**参数说明：**

name：图形文件名

x1, y1,x2, y2：窗口左上、右下坐标（可整体省略）

n：显示屏号（可省略）

**返回值：**无

**备注：**

1. 文件名中的目录必须使用“/”或“\\”分隔符，图形文件名支持绝对路径和相当路径，若是相对路径则以环境变量SEROOT\graph为当前路径，graph目录下的各级子目录名可以省略，默认扩展名.svm也可省略，例如以下几个例子均指向同一个文件：

* D:/Work/QT-ISCS/QT-ISCS/graph/mimicP/车站地理图.SVM
* mimicP/车站地理图.SVM
* 车站地理图.SVM或车站地理图

1. 只输入两个参数时，第二个参数x1的含义是重新打开图形标志，为0不重新打开已存在图形，默认为1强制重新打开图形
2. 若四个坐标参数省略或都为0则使用图形文件的默认坐标，四个坐标是一个整体，只有在同时输入时才有效
3. 显示屏号用于打开指定的屏幕上的图形窗口，屏号从1开始，若省略则打开脚本所在的屏幕上的图形窗口

##### 关闭图形窗口void closewnd (char \*name, int n)

**参数说明：**

name：图形文件名（规则同上）

n：显示屏号（可省略）

**返回值：**无

**备注：**显示屏号用于在关闭指定的屏幕上的图形窗口，屏号从1开始，若省略则关闭脚本所在的屏幕上图形窗口

1).如果有参数则将参数对应的svm窗口关闭.例如打开PA的脚本为openwnds("%GROUP%/通信/PA控制台")，关闭的脚本则为closewnd("%GROUP%/通信/PA控制台").

2).如果函数里面没有参数,则将当前活动的窗口关闭，closewnd().

##### 遥控void yk0 (char \* ObjectCode, char \* DESC,int Operation, int ExecMode)

参数说明：执行命令前弹出确认框

ObjectCode：遥控对象代码

DESC：确认框内显示的文字,框内完全跟该参数一致

Operation：遥控状态

ExecMode：执行模式（可省略）0：予置并执行1：直接执行（默认值）

执行逻辑：

1. RBAC\_Initialize()
2. RBAC\_CheckPrivilege(DEF\_RIGHT\_OPERATOR)—有无调度员权限

无调度员权限弹出“调度员对该点无控制权限！”提示框后退出函数。有权限则执行（3）。

1. CheckAoj(ObjectCode,p\_do,DEF\_RIGHT\_DEVICE\_C)---权限区域控制验证，返回TRUE则执行（4），否则直接退出函数。
2. 弹出二次确认框，框内文字为DESC，如果确认，执行（5）；如果取消，执行（6）。
3. 确认之后，判断ExecMode的值，

0则ExecMode=CSI\_K\_SELECT\_EXECUTE;

1则ExecMode =CSI\_K\_ONLY\_EXECUTE。然后执行DiscreteControl(user,ObjCode,ExecMode,Operation)，然后执行SetVdataInt(0,1)。

1. 执行SetVdataInt(0,0)。

##### 遥控void yk (char \* ObjectCode, int Operation, int ExecMode)

参数说明：

ObjectCode：遥控对象代码

Operation：遥控状态

ExecMode：执行模式（可省略）0：予置并执行；1：直接执行（默认值）

执行逻辑：

1. RBAC\_Initialize()
2. RBAC\_CheckPrivilege(DEF\_RIGHT\_OPERATOR)—有无调度员权限

无调度员权限弹出“调度员对该点无控制权限！”提示框后退出函数。有权限则执行（3）。

1. CheckAoj(ObjectCode,p\_do,DEF\_RIGHT\_DEVICE\_C)---权限区域控制验证，返回TRUE则执行（4），否则直接退出函数。
2. 判断ExecMode的值，

0则ExecMode=CSI\_K\_SELECT\_EXECUTE;

1则ExecMode =CSI\_K\_ONLY\_EXECUTE。然后执行DiscreteControl(user,ObjCode,ExecMode,Operation)。

##### 遥控void yk2 (char \* ObjectCode, char \* DESC,int Operation, int ExecMode)

参数说明：执行命令前弹出确认框

ObjectCode：遥控对象代码

DESC：确认框内显示的文字

Operation：遥控状态

ExecMode：执行模式（可省略）

0：予置并执行

1：直接执行（默认值）

执行逻辑：

* 1. RBAC\_Initialize()
  2. RBAC\_CheckPrivilege(DEF\_RIGHT\_OPERATOR)—有无调度员权限，无调度员权限弹出“调度员对该点无控制权限！”提示框后退出函数；有权限则执行（3）。
  3. CheckAoj(ObjectCode,p\_do,DEF\_RIGHT\_DEVICE\_C)-- 权限区域控制验证，返回TRUE则执行（4），否则直接退出函数。
  4. 数据库查找，select F1001\_GROUPCODE,F4009\_POINTCODE from SEMODEL.TB4009\_DIGPOINT where F4009\_POINTCODE=’ObjectCode’。如果查询不到则退出函数；如果查询到了，则用查到的GroupCode组成查询语句select F1001\_GROUPDESC from SEMODEL.TB1001\_GROUP where F1001\_GROUPCODE=’GroupCode’。如果查询不到则直接退出函数；查询到了，则记下查询到的GroupDesc，执行（5）。

1. 组成二次确认文字内容：是否执行+GroupDesc+DESC+？。然后弹出二次确认对话框，确认则判断ExecMode的值，

0则ExecMode=CSI\_K\_SELECT\_EXECUTE;

1则ExecMode =CSI\_K\_ONLY\_EXECUTE。然后执行DiscreteControl(user,ObjCode,ExecMode,Operation)，然后执行SetVdataInt(0,1)；取消则执行SetVdataInt(0,0)。

##### 遥调void yt0 (char \* ObjectCode, char \* DESC,int Operation, int ExecMode)

参数说明：执行命令前弹出确认框

ObjectCode：遥控对象代码

DESC：确认框内显示的文字,框内完全跟该参数一致

Operation：遥调状态

ExecMode：执行模式（可省略）

0：予置并执行

1：直接执行（默认值）

执行逻辑：

1. RBAC\_Initialize()
2. RBAC\_CheckPrivilege(DEF\_RIGHT\_OPERATOR)--- 有无调度员权限，无调度员权限弹出“调度员对该点无控制权限！”提示框后退出函数。有权限则执行（3）。
3. CheckAoj(ObjectCode,p\_ao,DEF\_RIGHT\_DEVICE\_C) -- 权限区域控制验证，返回TRUE则执行（4），否则直接退出函数。
4. 弹出二次确认框，框内文字为DESC，如果确认，执行（5）；如果取消，执行（6）。
5. 确认之后，判断ExecMode的值，

0则ExecMode=CSI\_K\_SELECT\_EXECUTE;

1则ExecMode =CSI\_K\_ONLY\_EXECUTE。然后执行TapControl(user,ObjCode,ExecMode,(float)Operation)，然后执行SetVdataInt(0,1)。

1. 执行SetVdataInt(0,0)。

##### 遥调void yt (char \* ObjectCode, float TapVal, int ExecMode)

参数说明：

ObjectCode：遥调对象代码

TapVal：遥调值

ExecMode：执行模式（可省略）

0：予置并执行

1：直接执行（默认值）

执行逻辑：

1. RBAC\_Initialize()
2. RBAC\_CheckPrivilege(DEF\_RIGHT\_OPERATOR)--- 有无调度员权限，无调度员权限弹出“调度员对该点无控制权限！”提示框后退出函数。有权限则执行（3）。
3. CheckAoj(ObjectCode,p\_ao,DEF\_RIGHT\_DEVICE\_C) -- 权限区域控制验证，返回TRUE则执行（4），否则直接退出函数。
4. 判断ExecMode的值，

0则ExecMode=CSI\_K\_SELECT\_EXECUTE;

1则ExecMode =CSI\_K\_ONLY\_EXECUTE。然后执行TapControl(user,ObjCode,ExecMode,TapVal)。

##### 遥调void yt2 (char \* ObjectCode, char \* DESC,int Operation, int ExecMode)

参数说明：执行命令前弹出确认框

ObjectCode：遥控对象代码

DESC：确认框内显示的文字

Operation：遥调值

ExecMode：执行模式（可省略）

0：予置并执行

1：直接执行（默认值）

执行逻辑：

1. RBAC\_Initialize()
2. RBAC\_CheckPrivilege(DEF\_RIGHT\_OPERATOR)--- 有无调度员权限，无调度员权限弹出“调度员对该点无控制权限！”提示框后退出函数。有权限则执行（3）。
3. CheckAoj(ObjectCode,p\_ao,DEF\_RIGHT\_DEVICE\_C) -- 权限区域控制验证，返回TRUE则执行（4），否则直接退出函数。
4. 数据库查找，select F1001\_GROUPCODE,F4007\_POINTDESC from SEMODEL.TB4007\_ANAPOINT where F4007\_POINTCODE=’ObjectCode’。如果查询不到则退出函数；如果查询到了，则用查到的GroupCode组成查询语句select F1001\_GROUPDESC from SEMODEL.TB1001\_GROUP where F1001\_GROUPCODE=’GroupCode’。如果查询不到则直接退出函数；查询到了，则记下查询到的GroupDesc，执行（5）。
5. 组成二次确认文字内容：是否执行+GroupDesc+DESC+？。然后弹出二次确认对话框，确认则判断ExecMode的值，

0则ExecMode=CSI\_K\_SELECT\_EXECUTE;

1则ExecMode =CSI\_K\_ONLY\_EXECUTE。然后执行TapControl(user,ObjCode,ExecMode,Operation)，然后执行SetVdataInt(0,1)；取消则执行SetVdataInt(0,0)。

##### 序控void trainctrl (char \*TrainCode)

参数说明：TrainCode:序控代码

执行逻辑：直接执行TrainControl(user,TrainCode,0,0)

##### 模式控制void modectrl (char \*GroupCode, int Mode)

参数说明：

GroupCode：组代码

Mode：模式号

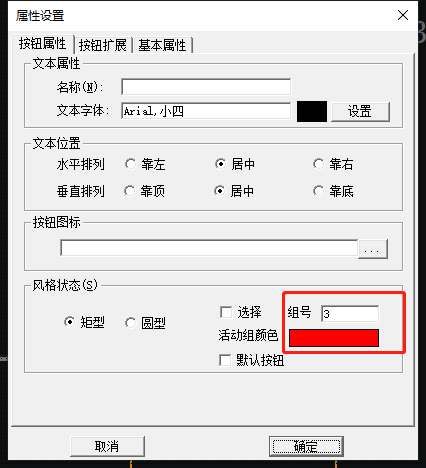
执行逻辑：直接执行SetGroupMode(user, GroupCode, Mode)。

##### 设置柱状图着色方案void setbarcolor (char \*obj, int r, int g, int b, int flag)

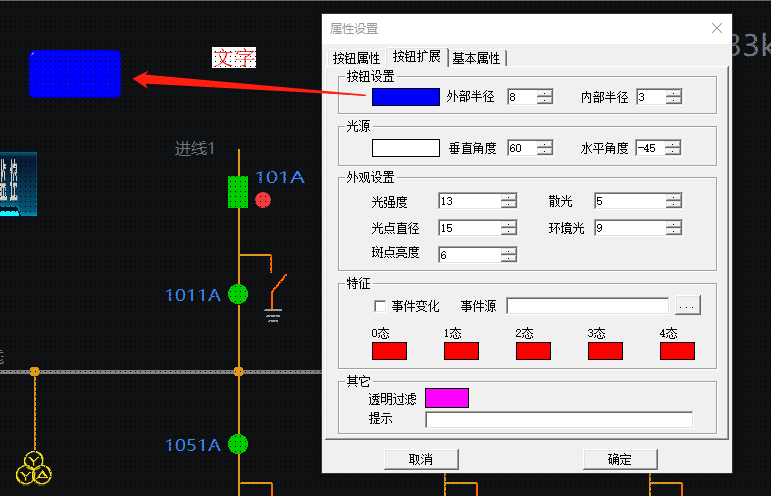
功能：根据obj（图元名称，如0\_1\_棒图\_813）改变图元的背景色（r，g，b），flag=0则不改变，1才改变。只能改变棒图的颜色。

##### 设置按钮图元为选中状态void setbtnsel (char \*obj, int group)

功能：将组号==group的按钮图元obj的边框颜色变成活动组设置的颜色，宽度为2。如果obj不存在就将组号==group的所有按钮设置为非选择态。



而按钮显示的颜色为ehgraph按钮拓展里的颜色，如下图



备注：组号相同的按钮，最多只能有一个被选中。

##### 设置按钮图元颜色void setbtncolor (char \*obj, int r, int g, int b)

功能：根据r，g，b设置按钮图元obj的按钮拓展里按钮设置的颜色（上图）。

##### 获取用户组角色名字串char\* getaojname ()

直接执行RBAC\_GetUserRoleDescription(NULL,NULL,NULL,szRoleName),szRoleName就是返回的角色名字符串。

##### 获取当前鼠标左键点击所在屏号int getscreenno ()

返回值：返回当前鼠标左键点击所在的显示屏逻辑号1至8（最多支持8屏）

##### 获取图元的外框大小 int getobjsize(char\* obj)

参数说明：

obj：图元名（图元脚本可省略）

返回值：图元外框的宽x2-x1和高y1-y2

例：A1,A2= getobjsize ("图元名")，分别将x2-x1和y2-y1的两个值赋值给A1、A2。

##### 获取图元的RGB颜色 int getobjcolor(char\* obj)

参数说明：

obj：图元名（图元脚本可省略）

返回值：图元的RGB值

例：A1,A2,A3=getobjcolor("图元名")，分别将R、G、B三个值赋值给A1、A2、A3变量

C/S端目前可获取的对象：文本、按钮。文本则返回字体颜色，按钮则返回ehgraph里按钮拓展里按钮设置的颜色，即显示的颜色。

##### 获取图元的图元名称 int getsymbolname()

说明： 获取当前图元名称，不带参数，图元脚本中使用。

返回值：图元的名称，如0\_1\_文本\_812。

##### 获取图形打开的当前屏号 int getscreenidx()

说明： 获取当前svm页面在第几屏幕打开的值，不带参数

返回值：图形打开的当前屏号

##### 获取图元当前坐标 int getobjpos (char\* obj)

说明： 获取图元的4个顶点坐标。

返回值：x1,y1,x2,y2的坐标值

例：x1,y1,x2,y2 = getobjpos("图元名")。

##### 移动图元位置void moveobj (char \*obj, int x, int y)

功能：将图元obj的中心点移动

x：x轴方向的位移 负数向左 正数向右

y：y轴方向的位移 负数向下 正数向上

##### 移动图元到指定位置void moveobjto (char \*obj, int x, int y)

功能：将图元obj的中心点移动到（x，y）位置。

##### 更改图元大小void changeobjsize (char \*obj, int x, int y)

功能：改变obj尺寸，

x：x轴方向的尺寸改变 负数变小 正数变大 左边固定右边改变

y：y轴方向的尺寸改变 负数变小 正数变大 下边固定上边改变

也就是从右边变大变小，从下边变大变小。

##### 更改图元线性变化void changeobjratio (char \*obj, int nx, int ny, int bx, int by)

功能：改变obj尺寸，

nx：设置x轴的宽

ny：设置y轴的高

bx：设置变化端 0--右端变化 1--左端变化 2--不变化

by：设置变化端 0—上端变化 1—下端变化 2--不变化

##### 更改图元角度void changeobjangle (char \*obj, int angle)

功能：改变图元角度，angle>0逆时针旋转，angle<0顺时针旋转。

##### 设置指定插件的参数 int setplugpara (plug\_name, para\_id, para, para\_len)

参数说明：只针对文本框插件

Plug\_name: 绘图时设定的插件名

Para\_id: 需要修改的插件参数ID

10001：对文本框清空再重新写入

10002：文本框清换行追加写入

Para: 具体的插件参数内容，字符串或者数字

Para\_len: 参数内容长度，字节数

返回值： 无

##### 获取控制返回结果int getctrlresult ()

返回值：返回两个字符串：控制对象的点代码PointCode和状态（“OK”或者“FAIL”）

如果没有获取到结果，则为null，需稍后再次执行。

示例：obj, status = getctrlresult()

功能逻辑：执行int nRet=ScadaReply(“”,&pRetBuf,&nRetLen)，nRet==CSI\_K\_PIPE\_NULL(无管道)或者pRetBuf==NULL或者nRetLen<=0就直接退出；如果nRet==CSI\_K\_PIPE\_DSCRTCTRLRESULT,说明返回的pRetBuf是遥控结果，“OK”或者“FAIL”。

##### 设置窗口背景色void setwndbkcolor (int r, int g, int b, int n)

**参数说明：**

r：红色值

g：绿色值

b：蓝色值

n：显示屏号（可省略）

**返回值：**无

**备注：**

1. 设置背景色配合打开、关闭窗口函数使用，减少窗口切换闪烁感
2. 显示屏号用于设置指定的屏幕的背景色，屏号从1开始，若省略则设置脚本所在的屏幕的背景色

##### 暂停功能void pause(int timeout)

功能：暂停界面timeout毫秒

**备注：**暂停功能用Sleep实现，在暂停过程中界面将会不响应鼠标操作，尽量在窗口脚本（异步执行）中使用

##### 简单睡眠 void sleep(milliseconds)

参数说明：

milliseconds: 需要睡眠的毫秒数

返回值： 无

说明： 实现简单的sleep功能，一般用于异步执行的脚本；与pause相比，不处理窗口的消息。也就是函数执行完毕后还是会响应界面操作。

##### 日志打印 int printlog(content)

参数说明：

content: 需要输出的内容，字符串

返回值： 无

说明： 便于调试或输出重要的日志，通过ehview查看【调度员界面】中的【script】。script要全小写，否则log输出不了。

##### 取得图元的着色颜色int getrealsetcolor (char \*obj)

功能：根据obj（图元名称，如0\_1\_文本\_813）得到图元setmetacolor()过后的着色标志（flag）和颜色（r，g，b），按顺序返回4个值：flag，r，g，b。如果这个图元obj之前没有setmetacolor()过，返回都为0。

##### 设置图元的着色颜色void setrealsetcolor (char \*obj, int r, int g, int b, int flag)

初步实验用起来跟setmetacolor作用一样，但工程部一般用setnetacolor。

##### 取得图元的刷新标志int getfreshflag (char \*obj)

功能：根据obj得到图元的刷新标志，0-不刷新；1-刷新。

##### 设置图元的刷新标志void setfreshflag (char \*obj, int flag)

这个刷新标志基本不用。