

# SLM 软件

---

SLM\_SDK2.1 开发者帮助文档

## 目录

Timeout_CreateWindow .....	3
Timeout_ShowWindow.....	4
Timeout_ShowImageFromFilePath.....	5
Timeout_CloseWindow.....	6
saShowImage FromFolder.....	7
saShowImage FormFolderPath.....	8
saShowImage FromSelector .....	9
saShowBlock .....	10
saShowWindow.....	11
saCloseWindow.....	12
saPauseShow .....	13
saResumeShow .....	14
saStopShow.....	15

## SLM\_SDK2.0 → SLM\_SDK2.1 升级简要说明

- 1、SLM\_SDK2.1 将提供 8 位灰度/24 位彩色图像的加载和显示。
- 2、SLM\_SDK2.1 将提供 20HZ 稳定的刷新送图接口。
- 3、SLM\_SDK2.1 新增 Timeout 函数组，用于开发时在循环代码中的送图功能。

Timeout\_ShowImagFromFilePath 函数可以单独显示一张图片，并且可以在显示后写入想要执行的代码。

## Timeout\_CreateWindow

`void Timeout_CreateWindow()`

**示例调用(C 代码):**

`Timeout_CreateWindow ()`

**说明:**

调用该函数将创建一个带延时属性的显示窗口。调用该函数并不会在桌面上显示一个窗口，但首次调用 `Timeout_ShowImageFromFilePath` 函数时，必须确保有这样一个窗口存在。要显示该窗口，请调用 `Timeout_ShowWindow(true)`。

使用该函数（Timeout 系列函数）创建的窗口和 sa 系列函数创建的窗口将是分别独立的窗口，因此若使用该函数创建 Timeout 窗口，请采用 Timeout 系列的窗口管理函数；若使用 sa 系列函数显示图片，请采用 sa 系列的窗口管理函数。

## Timeout\_ShowWindow

`void Timeout_CreateWindow(bool isShow)` //isShow:窗口是否显示

### 示例调用(C 代码):

```
Timeout_ShowWindow (ture);
```

### 说明:

该函数是 Timeout 系列函数的窗口管理函数。调用该函数将显示一个带有延时属性的窗口。如果该窗口存在，则显示该窗口；如果窗口不存在，则调用该函数不会有反应。但调用 Timeout\_ShowImageFromFilePath 显示图片时，必须确保有这样一个窗口存在，并已经显示出来。

### 示例调用(C 代码):

```
Timeout_ShowWindow (false);
```

### 说明:

该函数是 Timeout 系列函数的窗口管理函数。调用该函数将隐藏一个带有延时属性的窗口。如果该窗口存在，则隐藏该窗口；如果窗口不存在或已隐藏，则调用该函数不会有反应。

## Timeout\_ShowImageFromFilePath

```
void Timeout_ShowImageFromFilePath (  
char* szPath,           //图片文件路径  
bool IsStretch,         //是否拉伸。为 1 则拉伸图像铺满窗口 为 0 则不拉伸并在窗口内居中显示  
int DesX,               //显示区域的起点 x 坐标值(屏幕坐标系) (单位: pixel)  
int DesY,               //显示区域的起点 y 坐标值(屏幕坐标系) (单位: pixel)  
int DesWidth,           //显示区域的宽度 (单位: pixel)  
int DesHeight,          //显示区域的高度 (单位: pixel)  
bool IsRGB,             //当 IsRGB 为 true 时显示彩色图像  
int uMill)              //图片显示持续时间
```

### 示例调用(C 代码):

```
Timeout_ShowImageFromFilePath ("123.jpg",false,0,0,1920,1080,true,4000 ;
```

### 说明:

该函数是 Timeout 系列函数的显示函数。调用该函数可以将一张图片显示到延时显示的窗口中。UMill 参数是该图片的保持时间。如果仅显示一张图片或显示到最后一张图片，那么该图片维持显示状态，直到窗口关闭。如果要在for 循环里显示图片，那么在 uMill 时间流逝后，图片将切换成下一张图片。

注意：调用该函数前，必须首先调用 Timeout\_CreateWindow 和 Timeout\_ShowWindow 函数将窗口环境创建和显示出来。

注意：如果要通过调用该函数在循环代码里显示图片，请优先使用该函数，而不要使用 saShowImageForomFilePath 函数。

## Timeout\_CloseWindow

`void Timeout_CloseWindow()`

**示例调用(C 代码):**

```
Timeout_CloseWindow ();
```

**说明:**

该函数是 Timeout 系列函数的窗口控制函数。调用该函数关闭 Timeout 显示窗口。

**注意:** 请在主窗口退出前, 或进程终止前调用 Timeout\_CloseWindow 函数。如果不调用 Timeout\_CloseWindow, 那么尚处在循环中的 Timeout\_ShowImageFromFilePath 将不会立刻结束, 而直到循环代码跳出后才能停止 Timeout\_ShowImageFromFilePath 的送图行为。

## saShowImageFromFolder

```
void saShowImageFromFolder (
bool IsStretch,           //是否拉伸。为 1 则拉伸图像铺满窗口 为 0 则不拉伸并在窗口内居中显示
int DesX,                 //显示区域的起点 x 坐标值(屏幕坐标系) (单位: pixel)
int DesY,                 //显示区域的起点 y 坐标值(屏幕坐标系) (单位: pixel)
int DesWidth,             //显示区域的宽度 (单位: pixel)
int DesHeight,            //显示区域的高度 (单位: pixel)
int uMill,                //播放间隔 (单位: ms)
bool IsRGB)               //当 IsRGB 为 true 时显示彩色图像
```

### 示例调用(C 代码):

```
saShowImageFromFolder (0,1920,0,1920,1080,500,true);
```

### 说明:

调用该函数将弹出一个文件夹选择器，选中文件夹后点击确定，函数将以不拉伸方式将文件夹下的所有图片显示在由点{1920， 0}、{3840， 0}、1920， 1080}、{3840， 1080}围成的矩形区域内，并且每隔 500 毫秒切换到下一张图片。如果 IsRGB 参数为 true，则图像以原始色彩显示，如果 IsRGB 为 false，则图像以灰度方式显示。如果图片分辨率大于 1920x1080，那么图片将被矩形区域所裁剪。如果图片分辨率小于 1920x1080，图片将居中显示在矩形区域内。所支持的图片格式有：bmp、jpg、png、tif



文件夹择窗口

### 注意:

- 不拉伸方式将保证图片的还原度，不会失真。拉伸方式，则不论图片分辨率多大，都会铺满指定的显示区域。如果图片过度拉伸/压缩，将会损失清晰度造成失真。
- 不应该一次性将超过 3000 张图片送入，可能会造成系统资源耗尽而程序崩溃。
- 不应该将其他文件和图片混合放在一个文件夹下，可能会导致意想不到的错误
- 该函数不会读取文件夹下的子文件夹，所以，要显示的图片应该放在被选择的文件夹下，而不是放在它的子文件夹下面。



## saShowImageFormFolderPath

```
void saShowImageFromFolderPath (
char* FolderPath,           //文件夹路径
bool IsStretch,             //是否拉伸。为 1 则拉伸图像铺满窗口 为 0 则不拉伸并在窗口内居中显示
int DesX,                   //显示区域的起点 x 坐标值(屏幕坐标系) (单位: pixel)
int DesY,                   //显示区域的起点 y 坐标值(屏幕坐标系) (单位: pixel)
int DesWidth,               //显示区域的宽度 (单位: pixel)
int DesHeight,              //显示区域的高度 (单位: pixel)
int uMill,                  //播放间隔 (单位: ms)
bool IsRGB)                 //当 IsRGB 为 true 时显示彩色图像
```

### 示例调用(C 代码):

```
saShowImageFromFolderPath ("c:\\picture",0,1920,0,1920,1080,500,true);
```

### 说明:

调用该函数与 saShowImageFromFolder 功能类似，只是它不会弹出一个文件夹选择器，而是通过传入文件夹的路径参数来解析图片文件。函数将以不拉伸方式将文件夹下的所有图片显示在由点{1920, 0}、{3840, 0}、1920, 1080}、{3840, 1080}围成的矩形区域内，并且每隔 500 毫秒切换到下一张图片。**如果 IsRGB 参数为 true，则图像以原始色彩显示，如果 IsRGB 为 false，则图像以灰度方式显示。**如果图片分辨率大于 1920x1080，那么图片将被矩形区域所裁剪。如果图片分辨率小于 1920x1080，图片将居中显示在矩形区域内。所支持的图片格式有：bmp、jpg、png、tif

### 注意:

不拉伸方式将保证图片的还原度，不会失真。拉伸方式，则不论图片分辨率多大，都会铺满指定的显示区域。如果图片过度拉伸/压缩，将会损失清晰度造成失真。

不应该一次性将超过 3000 张的图片送入，可能会造成系统资源耗尽而程序崩溃。

不应该将其他文件和图片混合放在一个文件夹下，可能会导致意想不到的错误

该函数不会读取文件夹下的子文件夹，所以，要显示的图片应该放在被选择的文件夹下，而不是放在它的子文件夹下面。

# saShowImageFromSelector

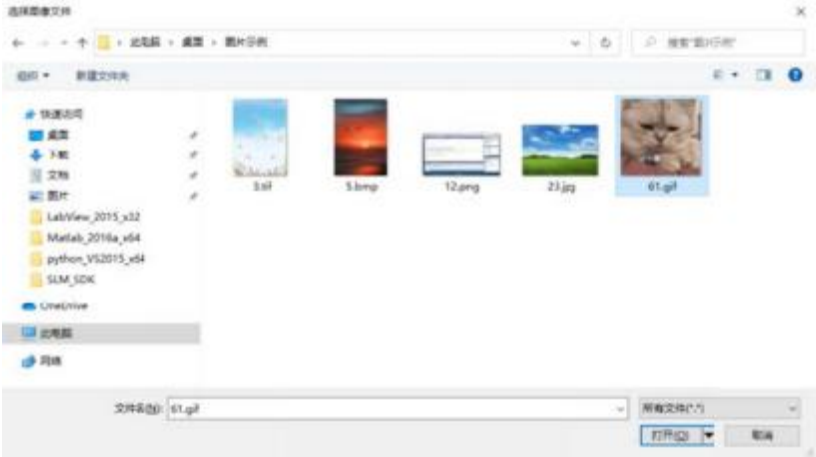
```
void saShowImageFromSelector (
bool IsStretch,           //是否拉伸。为 1 则拉伸图像铺满窗口 为 0 则不拉伸并在窗口内居中显示
int DesX,                 //显示区域的起点 x 坐标值(屏幕坐标系) (单位: pixel)
int DesY,                 //显示区域的起点 y 坐标值(屏幕坐标系) (单位: pixel)
int DesWidth,             //显示区域的宽度 (单位: pixel)
int DesHeight,            //显示区域的高度 (单位: pixel)
int uMill,                //播放间隔 (单位: ms)
bool IsRGB)                //当 IsRGB 为 true 时显示彩色图像
```

## 示例调用(C 代码):

```
saShowImageFromSelector (0,1920,0,1920,1080,500,true);
```

## 说明:

调用该函数将弹出一个文件过滤器，通过鼠标左键+ctrl 或 shift 自由挑选想要被加载的图片，点击确定后，函数将以不拉伸方式将所选择的图片显示在由点{1920, 0}、{3840, 0}、1920, 1080}、{3840, 1080}围成的矩形区域内，并且每隔 500 毫秒切换到下一张图片。**如果 IsRGB 参数为 true，则图像以原始色彩显示，如果 IsRGB 为 false，则图像以灰度方式显示。**如果图片分辨率大于 1920x1080，那么图片将被矩形区域所裁剪。如果图片分辨率小于 1920x1080，图片将居中显示在矩形区域内。所支持的图片格式有：bmp、jpg、png、tif



文件过滤器窗口

## 注意:

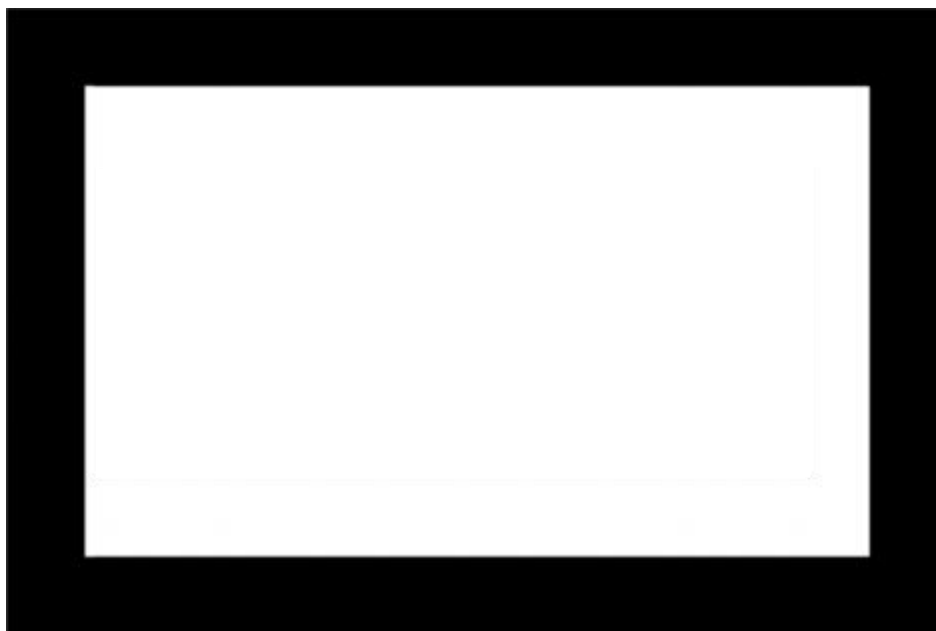
- 不拉伸方式将保证图片的还原度，不会失真。拉伸方式，则不论图片分辨率多大，都会铺满指定的显示区域。如果图片过度拉伸/压缩，将会损失清晰度造成失真。
- 不应该一次性将超过 3000 张图片送入，可能会造成系统资源耗尽而程序崩溃。
- 不应该选择非图片的文件，可能会导致意想不到的错误

## saShowBlock

```
void saShowBlock (
double* data,           //double 类型的数组
unsigned int data_cols, //数据块的宽度
unsigned int data_rows, //数据块的高度
int DesX,               //显示区域的起点 x 坐标值(屏幕坐标系) (单位: pixel)
int DesY,               //显示区域的起点 y 坐标值(屏幕坐标系) (单位: pixel)
int DesWidth,           //显示区域的宽度 (单位: pixel)
int DesHeight           //显示区域的高度 (单位: pixel)
bool IsRGB)             //当 IsRGB 为 true 时显示彩色图像
```

### 示例调用(C 代码):

```
double* data = new double[500*300];
for(int i=0;i<500*300;i++)
    data[i]=255;
saShowBlock (data,500,300,0 ,0,600 ,400,false);
delete data;
```



运行结果

### 说明:

调用该函数将 500x300 的数据块送入显示区域{0, 0}、{600, 0}、{0, 400}、{600, 400}。上图中白色区域是构造的 data 数据块, 黑色区域是函数参数指定的显示区域 0,0,600 ,400。

**利用此函数可以将任意构建的数据以图像形式呈现出来。**

## saShowWindow

```
void saShowWindow (  
bool IsShow) //布尔类型的参数 为 1 则显示 为 0 则隐藏
```

### 示例调用(C 代码):

```
saShowWindow(1); //如果显示区域已经被构建, 则显示这个显示区域  
saShowWindow(0); //如果显示区域已经被构建, 则隐藏这个显示区域
```

### 注意:

该函数是显示区域控制函数。

显示区域的构建只能由: saShowImageFromFilePath\_Timeout、saShowBlock 、saShowImageFromFolderPath、saShowImageFromFolder、saShowImageFromSelector 这 5 个函数来完成。如果显示区域没有被构建或者显示区域被撤销(撤销函数见 saCloseWindow),那么该函数不会有任何作用。

## saCloseWindow

void saCloseWindow ()

### 示例调用(C 代码):

```
saCloseWindow();           //如果显示区域 已经被构建， 则撤销这个显示区域
```

### 注意:

该函数是 **显示区域控制函数**。

显示区域的构建只能由: saShowImageFromFilePath、saShowBlock 、saShowImageFromFolderPath、saShowImageFromFolder、saShowImageFromSelector 这 5 个函数来完成。如果显示区域没有被构建或者显示区域被撤销,那么该函数不会有任何作用。

## saPauseShow

void saPauseShow ()

### 示例调用(C 代码):

```
saPauseShow();           //暂停一个连续播放图片的过程
```

### 注意:

该函数是播放控制函数。

如果一个连续播放的过程由 saShowImageFromFolderPath、saShowImageFromFolder、saShowImageFromSelector 创建，那么使用 saPauseShow 可以暂停这个播放过程。如果当前没有播放过程，该函数将不会起任何作用。

## saResumeShow

`void saResumeShow()`

### 示例调用(C 代码):

`saResumeShow();` //恢复一个由 saPauseShow 暂停的过程或当图片内容没有被切换时, 从头开始播放

### 注意:

该函数是 **播放控制函数**。

如果一个连续播放的过程由 saShowImageFromFolderPath、saShowImageFromFolder、saShowImageFromSelector 创建, 并且被 saPauseShow 暂停, 使用 ResumeShow 可以从中断的位置重新播放;

如果在第一次使用 saShowImageFromFolderPath、saShowImageFromFolder、saShowImageFromSelector 进行播放, 并且播放完成后没有使用上述 3 个函数加载新的图片, 那么可以直接使用 saResumeShow 从头开始一次播放。这主要是针对于加载大量图片设置的缓存优化功能。因为大量图片的加载是比较费时间的。

如果在第一次使用 saShowImageFromFolderPath、saShowImageFromFolder、saShowImageFromSelector 进行播放, 并使用 saStopShow 结束了播放, 那么调用 saResumeShow 将从头开始一次播放。

## saStopShow

`void saStopShow()`

**示例调用(C 代码):**

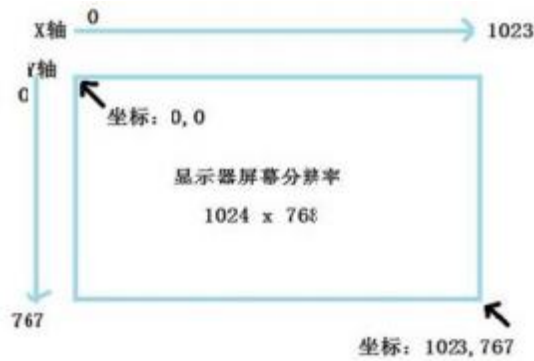
```
saStopShow();           //停止播放
```

停止一个已经开始的播放过程。



# 附录 A

屏幕坐标系和调制器屏幕在电脑上的设置



当插上调制器后，需要在系统里设置调制器屏幕的位置。



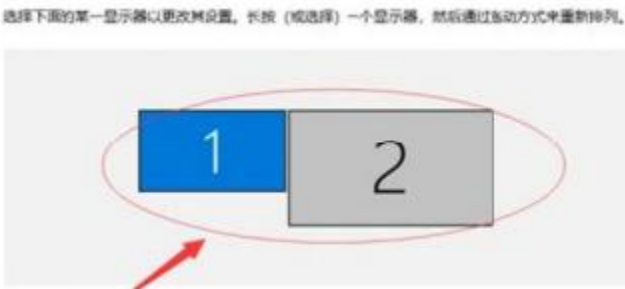
1、控制面板—系统



2、显示设置



3、选择“扩展这些显示器”



4、2#屏幕就是调制器屏幕

按照上面的设置步骤，如果计算机 1#屏的分辨率为 1920x1080，而调制器的分辨率为 2400x1600,那么在函数当中，要想在调制器上创建一个完全铺满调制器屏幕的显示区域，这个区域参数应该为 1920, 0, 2400, 1600。

## 附录 B—Demon 使用说明

Demon 共包含如下 6 个 Demon:

C#_VS2015_x64	2022/1/22 17:12	文件夹
C++_VS2015_x64	2022/1/22 16:31	文件夹
LabView_2015_x32	2022/1/21 16:48	文件夹
Matlab_2016a_x64	2022/1/22 17:10	文件夹
python_VS2015_x64	2022/1/23 0:17	文件夹
VB.net_VS2015_x64	2022/1/22 15:30	文件夹

1、C#打开 CallDemon\_WinForm.sln 运行即可。

本地磁盘 (C:) > Demon > C#\_VS2015\_x64 > CallDemon\_WinForm

名称	修改日期	类型	大小
.vs	2022/1/22 17:12	文件夹	
CallDemon_WinForm	2022/1/22 17:57	文件夹	
CallDemon_WinForm.sln	2022/1/22 17:12	Microsoft Visual ...	1 KB

2、C++打开 CallDemon.sln 运行即可

本地磁盘 (C:) > Demon > C++\_VS2015\_x64 > CallCemon

名称	修改日期	类型	大小
.vs	2022/1/22 15:31	文件夹	
CallDemon_Win32	2022/1/22 23:43	文件夹	
x64	2022/1/22 15:35	文件夹	
CallDemon.sdf	2022/1/22 23:45	SQL Server Com...	31,744 KB
CallDemon.sln	2022/1/22 17:08	Microsoft Visual ...	2 KB

3、LabView 打开 CallDemon.vi 运行即可

本地磁盘 (C:) > Demon > LabView\_2015\_x32

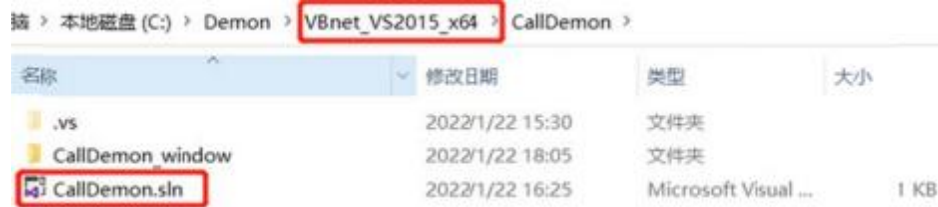
名称	修改日期	类型	大小
pg_10	2022/1/21 16:48	文件夹	
123.jpg	2022/1/19 17:38	JPG 文件	175 KB
CallDemon.vi	2022/1/21 16:58	LabVIEW Instru...	118 KB
comctl32.dll	2020/1/10 5:27	应用程序扩展	2,096 KB
comdlg32.dll	2020/1/10 5:27	应用程序扩展	795 KB
concr140.dll	2020/2/19 15:08	应用程序扩展	245 KB
concr140d.dll	2016/8/25 23:13	应用程序扩展	578 KB
SdiPlus.dll	2020/1/10 5:27	应用程序扩展	1,425 KB
nfc140.dll	2020/2/19 15:11	应用程序扩展	5,458 KB
nsimg32.dll	2020/1/10 5:27	应用程序扩展	8 KB
nsvcpl140.dll	2020/2/19 15:08	应用程序扩展	444 KB

4、Matlab 打开 CallDemon.m 文件，运行时按照 Matlab 软件提示，点击“更改文件夹”运行即可。

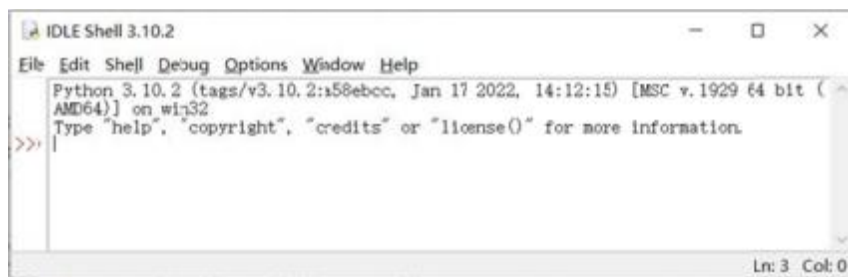
本地磁盘 (C:) > Demon > Matlab\_2016a\_x64

名称	修改日期	类型	大小
jpg_10	2022/1/21 13:59	文件夹	
123.jpg	2022/1/19 17:38	JPG 文件	175 KB
CallDemon.m	2022/1/21 15:07	M 文件	2 KB
comctl32.dll	2020/1/10 5:27	应用程序扩展	2,096 KB
comdlg32.dll	2020/1/10 5:27	应用程序扩展	795 KB
concr140.dll	2020/2/19 15:11	应用程序扩展	310 KB
concr140d.dll	2016/8/25 23:06	应用程序扩展	764 KB

5、VB 打开 CallDemon.Sln 文件运行即可



6、python 只提供了 4 个 py 文件，由于 SDK 的函数是按照并发设计的，所以直接运行 py 文件不会出现想要的结果。应该打开 python 的 IDLE 环境，或者 python 的命令行工具，将 py 文件里的代码放在里面运行。



Python IDLE 工具