* **API Reference**

L398界面：(可与E550/E560/L370/L398/M398共享)

[**PPL398\_InitialDevice**](#PPL398_InitialDevice)

[**PPL398\_InitialDeviceByWId**](#PPL398_InitialDeviceByWId)

[**PPL398\_UnInitialDevice**](#PPL398_UnInitialDevice)

[**PPL398\_OpenLCD**](#PPL398_OpenLCD)

[**PPL398\_Clear**](#PPL398_Clear)

[**PPL398\_GetHWPenID**](#PPL398_GetHWPenID)

[**PPL398\_GetHWPadID**](#PPL398_GetHWPadID)

[**PPL398\_GetHWVersionID**](#PPL398_GetHWVersionID)

[**PPL398\_DisplayHWVersion**](#PPL398_DisplayHWVersion)

[**PPL398\_SetPenStyle**](#PPL398_SetPenStyle)

[**PPL398\_SetPenWidth**](#PPL398_SetPenWidth)

[**PPL398\_SignatureStatusCallback**](#PPL398_SignPadEventCallback)

[**PPL398\_GetTotalPacketsNumber**](#PPL398_GetTotalInkPointsNumber)

[**PPL398\_getPackets**](#PPL398_GetInkPointData)

[**PPL398\_GetInkPointTime**](#PPL398_GetInkPointDataTimeInfo)

[**PPL398\_FreeInkPointTime**](#PPL398_FreeInkPointDataTimeInfo)

[**PPL398\_SaveDrawingImage**](#PPL398_SaveDrawingImage)

[**PPL398\_SetSaveCenterImageClip**](#PPL398_SetSaveClipImageMargin)

[**PPL398\_getDeviceInfo**](#PPL398_GetDeviceInfo)

[**PPL398\_AboutBox**](#PPL398_AboutBox)

[**PPL398\_AboutBoxByWId**](#PPL398_AboutBoxByWId)

[**PPL398\_PacketsBase64Encode**](#PPL398_Base64Encode)

[**PPL398\_PacketsBase64EncodeFree**](#PPL398_Base64EncodeFree)

[**PPL398\_Base64Decode**](#PPL398_Base64Decode)

[**PPL398\_GetTotalDecodePacketsNumber**](#PPL398_GetTotalDecodeInkPointsNumber)

[**PPL398\_GetDecodePacketsData**](#PPL398_GetDecodeInkPointData)

[PPL398\_SetPenColor](#PPL398_SetPenColor)

[PPL398\_RealTimeSignatureDataCallback](#PPL398_RealTimeSignatureDataCallback)

L398界面: (可与E550/E560/L370/L398/M398共享)

**PPL398\_InitialDevice**

设定L398签名板，初始化签名窗口以及设定签名画布大小。

bool PPL398\_InitialDevice(int nDeviceType, QWidget

\*pParentWidget, int nX, int nY, int nWidth, int nHeight)

**Parameter**

***nDeviceType***

[in] 设备类型ID。

设备ID定义：

enum DEVICE\_TYPE\_ID

{

PPL398\_DEVICE = 2

};

***pParentWidget***

[in] 父窗口指针。

***nX***

[in] 签名窗口起始点x坐标。

***nY***

[in] 签名窗口起始点y坐标。

***nWidth***

[in] 签名窗口宽度。

***nHeight***

[in] 签名窗口高度。

**Remark**

1. 可使用PPL398\_InitialDevice去设定画布大小。
2. 如需重新设定新画布大小时，必须先使用PPL398\_UnInitialDevice后，再使用PPL398\_InitialDevice设置一次。

**Return Value**

参阅回传值定义

**PPL398\_InitialDeviceByWId**

设定L398签名板，初始化签名窗口以及设定签名画布大小。

bool PPL398\_InitialDeviceByWId(int nDeviceType, WId

\*parentWId, int nX, int nY, int nWidth, int nHeight)

**Parameter**

***nDeviceType***

[in] 设备类型ID。

设备ID定义：

enum DEVICE\_TYPE\_ID

{

PPL398\_DEVICE = 2

};

***parentWId***

[in] 父窗口指针WId。

***nX***

[in] 签名窗口起始点x坐标。

***nY***

[in] 签名窗口起始点y坐标。

***nWidth***

[in] 签名窗口宽度。

***nHeight***

[in] 签名窗口高度。

**Remark**

1. 可使用PPL398\_InitialDeviceByWId去设定画布大小。
2. 如需重新设定新画布大小时，必须先使用PPL398\_UnInitialDevice后，再使用PPL398\_InitialDeviceByWId设置一次。

**Return Value**

参阅回传值定义

**PPL398\_UnInitialDevice**

卸除L398签名板，关闭签名窗口，释放资源。

int PPL398\_UnInitialDevice(int nDeviceType)

**Parameter**

***nDeviceType***

[in] 设备类型ID。

**Return value**

参阅回传值定义

**PPL398\_OpenLCD**

打开L398签名板LCD

int PPL398\_OpenLCD(int nDeviceType, bool bOn)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***bOn***

【打开LCD】：bOn = true

【关闭LCD】：bOn = false

**Return Value**

参阅回传值定义

**PPL398\_Clear**

清除L398签名板画布上的笔迹纪录。

int PPL398\_Clear (int nDeviceType)

**Parameter**

***nDeviceType***

[in] 设备类型ID

**Return value**

参阅回传值定义

**PPL398\_GetHWPenID**

获取L398签名板Pen ID

int PPL398\_GetHWPenID(int nDeviceType, char\* pPenID, int

nMaxCount)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pPenID***

[out] 输出Pen ID字符串指针

***nMaxCount***

[in] 输出Pen ID字符串的最大长度

**Return Value**

参阅回传值定义

**PPL398\_GetHWPadID**

获取L398签名板Pad ID

int PPL398\_GetHWPadID(int nDeviceType, char\* pPadID, int

nMaxCount)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pPadID***

[out] 输出Pad ID字符串指针

***nMaxCount***

[in] 输出Pad ID字符串的最大长度

**Return Value**

参阅回传值定义

**PPL398\_GetHWVersionID**

获取L398签名板的版本ID

int PPL398\_GetHWVersionID(int nDeviceType, char\* pVersion, int

nMaxCount)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pVersion***

[out] 输出版本ID字符串指针

***nMaxCount***

[in] 输出版本ID字符串的最大长度

**Return Value**

参阅回传值定义

**PPL398\_DisplayHWVersion**

开关L398签名板上版本信息的显示

Int PPL398\_DisplayHWVersion(int nDeviceType, bool bOn)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***bOn***

【显示】：bOn = true

【关闭】：bOn = false

**Return Value**

参阅回传值定义

**PPL398\_SetPenStyle**

设定L398签名板画笔种类。

int PPL398\_SetPenStyle (int nDeviceType, bool bPressureOn)

**Parameter**

***nDeviceType***

[in] 设备类型ID。

***bPressureOn***

[in] 画笔种类。

【压感笔】：bPressureOn = true

【定宽笔】：bPressureOn = false

**Return value**

参阅回传值定义

**PPL398\_SetPenWidth**

设定L398签名板画笔宽度。

int PPL398\_SetPenWidth (int nDeviceType, int nPenWidth)

**Parameter**

***nDeviceType***

[in] 设备类型ID。

***nPenWidth***

[in] 画笔宽度值。(1～10)

**Return value**

参阅回传值定义

**PPL398\_SignatureStatusCallback**

设定L398签名板上按【确定】或【取消】时的回传函数

LONG PPL398\_SignatureStatusCallback)(int nDeviceType,

NOTIFYSIGNSTATUS pfnSignatureStatusCallback)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pfnSignatureStatusCallback***

[in] 按【确定】或【取消】时的回传函数地址

回传函数定义：

typedef void (\*NOTIFYSIGNSTATUS)(int nEventID);

**Remark**

当L398签名板按【确定】或【取消】时会调用pfnSignatureStatusCallback。

**Return Value**

参阅回传值定义

**PPL398\_GetTotalPacketsNumber**

取得L398签名板目前笔迹数组点纪录数量。

int PPL398\_GetTotalPacketsNumber(int nDeviceType)

**Parameter**

***nDeviceType***

[in] 设备类型ID

**Return value**

回传笔迹数组点数量, 但如果笔迹点为空或者其他错误的话, 将会回传负的

错误值

**PPL398\_getPackets**

获取L398签名板单点笔迹数据

Int PPL398\_getPackets(int nDeviceType, PACKETS&

currentInkPoint)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***currentInkPoint***

[out] 笔迹数据结构体。

笔迹数据结构体:

typedef struct \_PACKETS

{

int nIndex;

short nX;

short nY;

short nPressureValue;

bool bStrokeEnd;

} PACKETS, \*PPACKETS;

**Return Value**

参阅回传值定义

**PPL398\_GetInkPointTime**

获取L398签名板单点笔迹的时间信息。

char\* PPL398\_GetInkPointTime(int nDeviceType, int nIndex)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nIndex***

[in] 单点笔迹序号

**Return value**

单点笔迹时间信息字符串。

**PPL398\_FreeInkPointTime**

释放L398签名板单点笔迹时间信息字符串。

void PPL398\_FreeInkPointTime(int nDeviceType, char\*

pInkPointTime)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pInkPointTime***

[in] 单点笔迹时间信息字符串

**Return value**

无。

**PPL398\_SaveDrawingImage**

储存L398签名板签名笔迹内容至支持的图像文件案。

int PPL398\_SaveDrawingImage(int nDeviceType, int

nImageTypeID, wchar\_t\* pImageFilePath, int

nDPILevel)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nImageTypeID***

[in] 储存影像文件类型

1 : BMP

2 : JPG

3 : PNG

4 : PDF

5 : SVG

***pImageFilePath***

[in] 储存影像文件名

***nDPILevel***

[in] 保存影像分辨率等级。(0～1)

0：150 DPI

1：300 DPI

**Remark**

支持的影像文件类型总共有BMP/JPG/PNG/PDF/SVG等五种格式，pImageFilePath为储存影像的完整文件名。

**Return value**

参阅回传值定义

**PPL398\_SetSaveCenterImageClip**

设定裁切L398签名板签名笔迹影像的水平和垂直边界。

int PPL398\_SetSaveCenterImageClip(int nDeviceType, bool

bClipImage, int nClipMarginX, int nClipMarginY)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***bClipImage***

[in] 是否要裁切笔迹影像?

【是】：bClipImage = true

【否】：bClipImage = false

***nClipMarginX***

[in] 裁切笔迹的同时, 并额外保留的水平边界。

***nClipMarginY***

[in] 裁切笔迹的同时, 并额外保留的垂直边界。

**Remark**

1) nClipMarginX，nClipMarginY要有作用，必须bClipImage等于Ture。

2)需搭配PPL398\_SaveDrawingImage API才能储存裁切笔迹影像。

**Return value**

参阅回传值定义

**PPL398\_getDeviceInfo**

获取L398签名板相关信息

int PPL398\_getDeviceInfo(int nDeviceType, int nIndex, void\*

pOutput)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nIndex***

[in] 设备信息序号

1 ：设备当前状态

2 ：签名设备的X的坐标范围

3 ：签名设备的Y的坐标范围

4 ：签名设备的最大压力值

5 ：签名设备的厂商名字

6 ：签名设备的设备名字

7 ：签名设备的SDK版本

***pOutput***

[out] 获取的设备信息

**Return Value**

参阅回传值定义

**PPL398\_AboutBox**

显示SDK版本信息。

void PPL398\_AboutBox(QWidget \*pParent)

**Parameter**

***pParent***

[in] 父窗口指针。

**Return value**

无。

**PPL398\_AboutBoxByWId**

显示SDK版本信息。

void PPL398\_AboutBoxByWId(WId parentWId)

**Parameter**

***pParent***

[in] 父窗口指针WId。

**Return value**

无。

**PPL398\_PacketsBase64Encode**

将L398签名板签名笔迹以指定的影像格式转换为Base64编码的字符串。

char\* PPL398\_PacketsBase64Encode (int nDeviceType, int nImageTypeID)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nImageTypeID***

[in] 欲编码的影像格式。

1 : BMP

2 : JPG

3 : PNG

4 : INK

**Return value**

回传编码后的字符串, 若失败会回传NULL

**PPL398\_PacketsBase64EncodeFree**

释放Base64编码占用的内存，删除编码数据。

void PPL398\_PacketsBase64EncodeFree(int nDeviceType, char\* pData)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pData***

[in] Base64编码数据指针

**Return value**

无

**PPL398\_Base64Decode**

将Base64编码的字符串译码为原始笔迹数据。

int PPL398\_Base64Decode(int nDeviceType, int nImageTypeID,

const char\* pEncodedData, wchar\_t\* pImageFilePath)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nImageTypeID***

[in] 编码的影像格式。

1 : BMP

2 : JPG

3 : PNG

4 : INK

***pEncodedData***

[in] 编码的数据字符串

***pImageFilePath***

[in] 译码数据储存完整文件名

**Return Value**

参阅回传值定义

**PPL398\_GetTotalDecodePacketsNumber**

获取解碼后得到的笔迹点数。

int PPL398\_GetTotalDecodePacketsNumber (int nDeviceType)

**Parameter**

***nDeviceType***

[in] 设备类型ID

**Return value**

回传笔迹数组点数量, 但如果笔迹点为空或者其他错误的话, 将会回传负的

错误值

**PPL398\_GetDecodePacketsData**

获取译码后得到的单点笔迹数据。

int PPL398\_GetDecodePacketsData (int nDeviceType, int nIndex,

int nTypeID)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nIndex***

[in] 笔迹点在数组中的序号

***nTypeID***

[in] 取值种类。

1：取得X坐标

2：取得Y坐标

3：取得是否为每笔划结束点状态

**Return Value**

参阅回传值定义

**PPL398\_SetPenColor**

设定L398画笔颜色。

int PPL398\_SetPenColor (int nDeviceType, int nR, int nG, int nB)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***nR***

[in] 画笔红色比例值。 (0~255)

***nG***

[in] 画笔绿色比例值。 (0~255)

***nB***

[in] 画笔蓝色比例值。 (0~255)

**Return value**

参阅回传值定义

**PPL398\_RealTimeSignatureDataCallback**

设定获取L398签名板上签名实时数据的回传函数

bool PPL398\_RealTimeSignatureDataCallback)(int nDeviceType,

NOTIFYREALTIMESIGNDATA pfnRealTimeSignDataCallback)

**Parameter**

***nDeviceType***

[in] 设备类型ID

***pfnRealTimeSignDataCallback***

[in] 获取L398签名板上签名实时数据的回传函数地址

回传函数定义：

typedef void (\*NOTIFYREALTIMESIGNDATA)(WPARAM wParam, LPARAM lParam);

**Remark**

L398签名板上签名实时数据会调用pfnSignatureStatusCallback。

**Return Value**

参阅回传值定义

**回传值:**

|  |  |  |
| --- | --- | --- |
| 定义名称 | 定义值 | 描述 |
| ARI\_SUCCESS | 0 | 成功 |
| ARI\_FAIL | -1 | 失败 |
| ARI\_INVALID\_PARAMETERS | -2 | 传入参数错误 |
| ARI\_DRIVER\_INIT\_FAIL | -3 | 【驱动】初始化失败 |
| ARI\_DRIVER\_OPEN\_DEVICE\_FAIL | -4 | 【驱动】打开设备失败 |
| ARI\_DRIVER\_CLAIM\_INTERFACE\_FAIL | -5 | 【驱动】宣告接口失败 |
| ARI\_DRIVER\_CREATE\_THREAD\_FAIL | -6 | 【驱动】建立Thread失败 |
| ARI\_DRIVER\_SENT\_CMD\_FAIL | -7 | 【驱动】下命令失败 |
| ARI\_INK\_EMPTY | -8 | 笔迹数组点为空 |