

ACLLib Document

目录

ACCLib	1
1. 主函数.....	1
2. 窗口.....	2
2.1 启动图形窗口.....	2
2.2 启动终端窗口.....	2
3. 绘制.....	2
3.1 开始/结束绘制	2
3.2 几何形状.....	2
3.2.1 线条.....	2
3.2.2 图形.....	4
3.3 画笔.....	5
3.4 画刷.....	5
3.5 文字.....	5
3.6 图片.....	6
3.6.1 加载.....	6
3.6.2 绘制.....	6
4. 声音.....	6
4.1 加载.....	6
4.2 播放.....	6
4.3 停止.....	7
5. 事件.....	7
5.1 事件回调函数.....	7
5.2 鼠标.....	7
5.3 键盘.....	7
5.4 字符输入.....	7
5.5 定时器.....	8
6. 光标.....	8

ACCLib

Advanced C Lab Library.

1.初始化函数

➤ `int Setup();`

ACCLib 程序运行开始时被调用的函数。

2. 窗口

2.1 启动图形窗口

➤ `void initWindow(const char title[], int left, int top, int width, int height);`

说明:

初始化程序窗口。该函数必须在 `Setup` 函数中最先调用, 并且只能调用一次。

参数:

`title` : 窗口标题

`left` : 窗口左上角横坐标, 若不希望指定窗口位置, 可传入 `DEFAULT`。

`top` : 窗口左上角纵坐标, 若不希望指定窗口位置, 可传入 `DEFAULT`。

`width` : 窗口可绘制区域的宽度。

`height` : 窗口可绘制区域的高度。

➤ `int getWidth();`

➤ `int getHeight();`

说明:

返回窗口绘图区域的宽/高。

2.2 启动终端窗口

➤ `void initConsole ();`

说明:

初始化终端窗口。在该函数执行后, 才能使用 `printf` 和 `scanf`。

参数:

无。

3. 绘制

3.1 开始/结束绘制

➤ `void beginPaint();`

➤ `void endPaint();`

说明:

`beginPaint()` 函数负责初始化绘图操作。3.2 - 3.6 中, 所有实际绘制图形的函数, 均应在调用 `beginPaint()` 函数后调用。绘制结束后, 调用 `endPaint()` 函数后, 结束绘图操作, 并且将刚才绘制的图形显示到屏幕上。

`beginPaint()` 与 `endPaint()` 必须成对使用, 每一个 `beginPaint` 调用均应当对应一个 `endPaint` 调用。建议在同一函数中完成一组绘图操作, 以防止函数调用不匹配。若未成对使用这一组函数, 如: 未首先调用 `beginPaint`, 便直接调用 `endPaint` 函数; 或者第一次调用 `beginPaint` 后, 未调用 `endPaint`, 又再次调用 `beginPaint` 函数。这些情况下程在运行时会弹出窗口提示错误。

绘图区域左上角为原点, 向右为 `x` 轴正方向, 向下为 `y` 轴正方向。

3.2 几何形状

3.2.1 线条

在绘图区域中, 有一个初始位置为(0,0)的绘图点。该点与线条绘制相关。

➤ `int getX(void);`

➤ `int getY(void);`

获取绘图点的 x、y 坐标。

➤ `void moveTo(int x, int y);`

➤ `void moveRel(int dx, int dy);`

移动绘图点。moveTo 将绘制点移动到屏幕坐标的对应位置，moveRel 将绘制点作(dx,dy)的相对位移。

➤ `void arc(int nLeftRect, int nTopRect, int nRightRect, int nBottomRect, \`
`int nXStartArc, int nYStartArc, int nXEndArc, int nYEndArc);`

说明：

绘制一段圆（椭圆）弧。

前 4 个参数给出一个矩形的左上角、右下角定点。绘制的弧线内切该矩形。后两个参数指定两个点，这两个点到矩形中心的连线作为圆弧起始、终止位置的线。

该函数的参数与 chrod、pie 相同，建议实际测试了解其使用方法。

➤ `void line(int x0, int y0, int x1, int y1);`

➤ `void lineTo(int x, int y);`

➤ `void lineRel(int dx, int dy);`

说明：

绘制直线。

line 函数直接绘制一条从(x0,y0)到(x1,y1)的直线，与绘制点的位置无关。

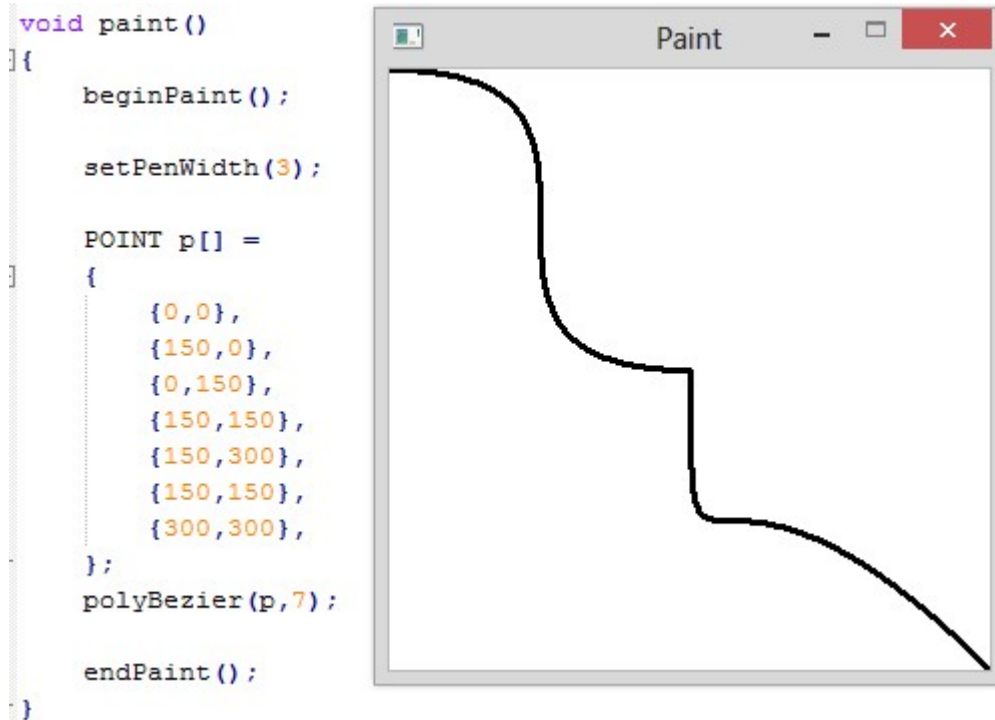
lineTo 从当前绘图点位置到屏幕坐标(x,y)绘制一条直线。lineRel 函数从当前绘图点位置到相对位移(dx,dy)的位置绘制一条直线。

➤ `void polyBezier(const POINT *lppt,int cPoints);`

说明：

绘制一条贝塞尔曲线。贝塞尔曲线的每个节点有两个控制点（两个端点各有一个控制点）。若需要绘制一条 n

个节点的贝塞尔曲线，需要传入 n*3-2 个点。数组中点的顺序应为：起点、起点控制点、第 2 个点的第一个控制点、第二个节点、第 2 个点的第二个控制点、……、终点的控制点、终点。



- `void polyline(const POINT *lppt, int cPoints);`

说明:

顺次连接传入的数组中所有的点。

3.2.2 图形

- `void chrod(int nLeftRect, int nTopRect, int nRightRect, int nBottomRect, \`
`int nXRadial1, int nYRadial1, int nXRadial2, int nYRadial2);`

说明:

绘制一块弓形，参数与 Arc 相同，建议实践测试效果。

- `void ellipse(int nLeftRect, int nTopRect, int nRightRect, int nBottomRect);`

说明:

绘制一个椭圆，参数为外切椭圆的矩形的左上角、右下角坐标。

- `void pie(int nLeftRect, int nTopRect, int nRightRect, int nBottomRect, \`
`int nXRadial1, int nYRadial1, int nXRadial2, int nYRadial2);`

说明:

绘制一块扇形，参数与 Arc 相同，建议实践测试效果。

- `void polygon(const POINT *lpPoints, int nCount);`

说明:

将输入数组中的所有点顺序连接，绘制一个多边形。

- `void rectangle(int nLeftRect, int nTopRect, int nRightRect, int nBottomRect);`

说明:

绘制一个矩形，参数为矩形左上角、右下角坐标。

➤ `void roundrect(int nLeftRect, int nTopRect, int nRightRect, int nBottomRect, int nWidth, int nHeight);`

说明:

绘制一个圆角矩形。最后两个参数为圆角部分的宽、高。

3.3 画笔

画笔对应所有图形边框的绘制。

画笔颜色可以使用头文件中预定的颜色，也可以使用 `RGB(r,g,b)`宏自行设定颜色，每种颜色分量的范围为 0-255。`setPenColor` 可以接受 `EMPTY` 作为参数，将线条颜色设置为透明。

参见 `acllib.h` 中 `ACL_Pen_Style`。

➤ `void setPenColor(ACL_Color color);`
➤ `void setPenWidth(int width);`
➤ `void setPenStyle(ACL_Pen_Style style);`

3.4 画刷

画刷对应所有图形填充颜色的绘制。画刷颜色也可以使用

用 `setBrushColor(EMPTY)`设置为透明。参见 `acllib.h` 中

`ACL_Brush_Style`。

➤ `void setBrushColor(ACL_Color color);`
➤ `void setBrushStyle(ACL_Brush_Style style);`

3.5 文字

设置文字的前景、背景、尺寸、字体。

➤ `void setTextColor(ACL_Color color);`
➤ `void setTextBkColor(ACL_Color color);`
➤ `void setTextSize(int size);`
➤ `void setTextFont(char *pFontName);`

➤ `void paintText(int x, int y, const char *pStr);`

说明:

将 `pStr` 所指向的文字输出到屏幕坐标 `x`、`y` 处。

3.6 图片

3.6.1 加载

➤ `void loadImage(const char *pImageFileName, ACL_Image *pImage);`

说明：

从文件中加载文件，仅支持 BMP、JPEG、GIF 格式。 第一个参数为文件名，第二个参数为指向一个 ACL_Image 结构的指针。

3.6.2 绘制

➤ `void putImage(ACL_Image *pImage, int x, int y);`

将图像绘制到点(x,y)处。

➤ `void putImageScale(ACL_Image *pImage, int x, int y, int width, int height);`

将图像绘制到点(x,y)处，并且改变图像的宽、高。

➤ `void putImageTransparent(ACL_Image *pImage,
int x, int y, int width, int height, ACL_Color bkColor);`

将图像绘制到点(x,y)处，可以改变图像的宽、高，最后一个参数中给出的颜色将被过滤为透明色。

4. 声音

4.1 加载

➤ `void loadSound(char *fileName, ACL_Sound *pSound);`

从文件中加载声音文件，第一个参数为文件名，第二个参数为指向一个 ACL_Sound 结构的指针。

4.2 播放

➤ `void playSound(ACL_Sound soundID, int repeat);`

播放声音，第二个参数非零时，音乐将循环播放。否则只播放一遍。多次调用该函数，可以同时播放多个声音。

4.3 停止

- `void stopSound(ACL_Sound soundID);`
停止播放给定的声音。

5. 事件

5.1 事件回调函数

当有输入事件发生时，系统会调用设定的回调函数。

5.2 鼠标

- `typedef void (*MouseEventCallback) (int x, int y, int button, int event);`
- `void registerMouseEvent(MouseEventCallback callback);`
前两个参数为鼠标指针的位置，`button` 为按键，`event` 为发生的事件（按下或弹起），鼠标移动时产生 `move` 事件。参见 `acllib.h` 中 `ACL_Mouse_Button`、`ACL_Mouse_Event`。

5.3 键盘

- `typedef void (*KeyboardEventCallback) (int key, int event);`
- `void registerKeyboardEvent(KeyboardEventCallback callback);`
第一个参数为按键的虚拟键码（注意：不是 ASCII 码）。第二个参数为发生的事件（按下或弹起。参见 `acllib.h` 中 `ACL_Key_Event`。
虚拟键码，参见：
<http://msdn.microsoft.com/en-us/library/windows/desktop/dd375731%28v=vs.85%29.aspx>。
或
`WinUser.h`

5.4 字符输入

- `typedef void (*CharEventCallback) (int key);`
- `void registerCharEvent(CharEventCallback callback);`
当按键产生文字输入时，出发此事件，参数为字符 ASCII 码，或对应中文字符编码。

5.5 定时器

- `void startTimer(int timerID, int timeinterval);`
- `void cancelTimer(int timerID);`
- `typedef void (*TimerEventCallback) (int timerID);`
- `void registerTimerEvent(TimerEventCallback callback);`

定时器时间间隔单位为毫秒。设置定时器后，每经过 `timeinterval` 毫秒时间，便会触发 `timer` 时间。可以使用多个 `timer`，ID 建议使用从 0 开始的整数。

当 `timeinterval` 设置为 0 时，定时器会在程序空闲时（没有键盘、鼠标输入，且上次 `timer` 事件已经处理）触发。

`timer` 的时间精度在 10ms 数量级，当设置的时间间隔小于 10ms 时，实际间隔会在 10ms 以上。

6. 光标

- `void setCaretSize(int w, int h);`
- `void setCaretPos(int x, int y);`
- `void showCaret();`
- `void hideCaret();`

说明：

这里的光标指的是输入文字时，闪烁指示当前输入位置的光标。输入文字时，光标并不会自动移动，需要手动调整光标位置。光标默认位置在窗口右下角，输入法的输入窗口会出现在光标所在位置。