Free Pascal从零开始编游戏

Display单元库教程

制作：ax_pokl

日期：2017-12-07

目录

前言

- 献给所有热爱游戏编程的程序员们。
- 建议Pascal初学者在阅读本教程之前先阅读《一天学会Free Pascal》教程。
- 本教程使用Windows NT为内核的操作系统。
- 本教程的实例均通过Free Pascal 3.0.0编译。
- 学习C语言的同学可以阅读本教程的姐妹版《C++从零开始编游戏》教程。

前言

- 本教程使用Display单元库，请先阅读单元库display.pp内所有文字说明和所有子程序重载
- 本教程所有实例永久下载地址：

http://axpokl.com/display.zip

http://axpokl.ys168.com/

- 作者ax_pokl联系方式：

E-mail：ax_pokl@sina.com，QQ：395838203。

- 由于作者水平有限，教程难免有错误和疏漏之处，敬请谅解。发现错误请联系作者，谢谢！

第一章 配置

- 第一节 Windows操作系统
- 第二节 Free Pascal编译器
- 第三节 Free Pascal IDE乱码问题
- 第四节 Display单元库

第一节 Windows操作系统

- Windows操作系统是微软公司推出的操作系统。正如其名，通过此操作系统可以建立窗口。
- 本教程所用的单元库Display使用了Windows API建立窗口并使用GDI+进行绘图，因此本教程只适用于Windows操作系统。请确保已经安装了以Windows NT为内核的操作系统。
- 您可以在编译时加入指令-WG，或者在代码头加入{$APPTYPE GUI}创建窗体应用程序。

第二节 Free Pascal编译器

- 为了编译Pascal语言程序，请下载并安装Free Pascal编译器（及其IDE）：

http://www.freepascal.org/down/i386/win32.var

- 官方手册：

http://www.freepascal.org/docs-html/fpctoc.html

- 本教程的编译器以Free Pascal Compiler version 3.0.0 for i386（fpc.exe）为准。

第二节 Free Pascal编译器

- 要编译Display单元库，请先配置环境变量。
- 您可在cmd.exe中输入以下命令添加环境变量：

SET PATH=%PATH%;[fpc.exe目录绝对路径]

- 您可以使用以下命令编译Display单元库：

fpc [display单元库目录绝对路径]/disp/display.pp

- 你可以使用Free Pascal IDE(fp.exe)或任意一款你喜欢的IDE（例如notepad.exe）书写并编译代码。

第三节 Free Pascal IDE乱码问题

- 请进行以下操作解决IDE(fp.exe)的乱码问题：
- 1、运行cmd.exe，输入以下命令并执行：

```
set key=HKCU\Console\^%SystemRoot^%_system32_cmd.exe
REG ADD %key% /v CodePage /t REG_DWORD /d 65001 /f
REG ADD %key% /v FaceName /t REG_SZ /d "Lucida Console“ /f
```

2、运行notepad.exe，输入以下代码并保存为fp.cmd：

```
chcp 437 & start /b /i /wait fp.exe
```

- 3、将fp.cmd剪切到fp.exe所在的目录并运行。

第四节 Display单元库

- 本教程使用Display单元库实现游戏：

http://axpokl.com/display.zip

http://axpokl.ys168.com

- 编译单元库源码Display.pp即可获得单元库的目标文件Display.o和编译库文件Display.ppu。
- 请将Display.pp（或编译好的Display.o及Display.ppu）拷贝到主程序同一个目录下。
- Display单元库仍在不断更新，每个版本并不互相兼容，因此对于每一个程序应使用独立的Display单元库。

第四节 Display单元库

- 引入后，可以使用Display单元库的子程序：

```
uses Display;//使用Display单元库
begin
Msgbox(i2s(GetError()));//弹出错误代码
end.
```

- GetError获取最后一个Windows API错误代码。
- i2s将整型转换为ansistring型。
- Msgbox弹出ansistring型窗口文字。

第二章 窗口

- 第一节 建立窗口
- 第二节 设定窗口标题
- 第三节 判断窗口状态
- 第四节 获取窗口大小
- 第五节 关闭窗口

第一节 建立窗口

- 以下过程可以创建窗口：

```
procedure CreateWin();
procedure CreateWin(c:longword);
procedure CreateWin(w,h:longword);
procedure CreateWin(w,h:longword;cfg,cbg:longword);
```

- 其中w,h代表宽度和高度，如不指定w,h则默认使用屏幕一半宽高来建立窗口。c为默认颜色。
- 颜色c为四字节ABGR模式（A=透明，B=蓝，G=绿，R=红，各占一个字节）。

第一节 建立窗口

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
Msgbox('窗口已建立');//输出窗口建立信息
end.
```

第二节 设定窗口标题

- 以下子程序可以设定或获取标题：

```
procedure SetTitle(s:ansistring);
function GetTitle():ansistring;
```

- 在Free Pascal中，如果实参为string类型，则调用子程序的时候类型会自动转为ansistring。

第二节 设定窗口标题

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
SetTitle('我是标题');//设定窗口标题
Msgbox(GetTitle());//获取并输出窗口标题
end.
```

第三节 判断窗口状态

- 以下函数可以判断窗口状态：

function IsWin():boolean;

- 如果窗口存在，则该函数返回true，否则返回false。

第三节 判断窗口状态

- 实例：

```
uses Display;//使用Display单元库
begin
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
CreateWin();//建立窗口
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
end.
```

第四节 获取窗口大小

- 以下函数可以获取窗口大小：

```
function GetWidth():longword;
function GetHeight():longword;
function GetSize():longword;
```

- 其中，GetSize的前两字节为宽，后两字节为高。可以用Hi(GetSize())和Lo(GetSize())获取。
- 如需改变窗口大小，需要使用消息传递函数。这会在教程的第二部分进行解说。

第四节 获取窗口大小

- 此外还有以下函数可以获取屏幕大小：

```
function GetScrWidth():longint;
function GetScrHeight():longint;
function GetScrSize():longword;
```

- 以及以下函数可以获取窗口位置：

```
function GetPosX():longint;
function GetPosY():longint;
```

第四节 获取窗口大小

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
Msgbox(i2s(GetWidth())+' '+i2s(GetHeight()));//输出窗口大小信息
end.
```

第五节 关闭窗口

- 以下过程可以关闭窗口：

```
procedure CloseWin();
```

- CloseWin过程不仅会关闭窗口，还会释放窗口句柄及设备上下文句柄（HDC）。也就是说，无法再用CreateBMP函数获取窗口内容，也无法使用LoadBMP函数读取图片。

第五节 关闭窗口

- 实例：

```
uses Display;//使用Display单元库
begin
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
CreateWin();//建立窗口
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
CloseWin();//关闭窗口
```

第五节 关闭窗口

```
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
CreateWin();//再次建立窗口
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
CloseWin();//再次关闭窗口
if IsWin() then Msgbox('Yes') else Msgbox('No');//输出窗口状态信息
end.
```

第三章 绘图

- 第一节 刷新窗口
- 第二节 绘制图形
- 第三节 读取图片
- 第四节 绘制图片
- 第五节 绘制拉伸图片
- 第六节 绘制透明图片
- 第七节 快速画点

第一节 刷新窗口

- 以下函数可以刷新窗口：

procedure FreshWin();

- 绘图完毕后须刷新窗口才能使绘制的内容生效（默认情况下，绘图子程序会绘制到缓冲区）。
- 使用绘图子程序会占用CPU，因此应尽量避免使用或减少使用次数，例如用图片代替图形。
- 刷新窗口后，帧率会自动更新。详情请阅读第五章第四节帧率获取。

第一节 刷新窗口

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin(Red);//建立红色窗口
Clear(Blue);//清屏蓝色
Msgbox('清屏没有生效');//这里清屏不会生效
FreshWin();//刷新窗口
Msgbox('清屏已生效');//这里清屏生效了
end.
```

第二节 绘制图形

- 以下过程可以绘制图形：

```
procedure SetPixel(x,y:longword;c:longword);
procedure Line(x,y:longword;w,h:longint;c:longword);
procedure Bar(x,y:longword;w,h:longint;c:longword);
procedure Circle(x,y,r:longint;c:longword);
procedure Ellipse(x,y,rx,ry:longint;c:longword);
procedure Clear();
procedure Clear(c:longword);
```

- Clear过程调用了Bar过程。

第二节 绘制图形

- 实例：

```
uses Display;//使用Display单元库
var n:longword=$1000;//绘制点数量
begin
CreateWin();//建立窗口
Line(10,10,100,100,Red);//绘制直线
Bar(110,10,100,100,Blue);//绘制矩形
Circle(60,160,50,Green);//绘制圆形
Ellipse(185,160,25,50,Pink);//绘制椭圆
```

第二节 绘制图形

```
while n>0 do
begin
SetPixel(random(GetWidth()),random(GetHeight()),random($FFFFFF));//随机画点
n:=n-1;
end;
FreshWin();//刷新窗口
Msgbox('绘制完成');//输出绘制完成信息
end.
```

第三节 读取图片

- 读取图片之前，请先创建pbitmap类型变量。
- pbitmap和bitmap类型结构：

```
type pbitmap=^bitmap;
type bitmap=record
Handle:longword;DC:longword;
Width:longword;Height:longword;
Color:longword;FileName:string;
end;
```

第三节 读取图片

- 以下函数可以读取图片：

function LoadBMP(s:ansistring):pbitmap;

function LoadBMP(s:ansistring;c:longword):pbitmap;

- s为文件名，c为图片背景颜色（默认透明色）。
- 函数返回pbitmap类型的图片。
- 支持的格式：BMP，PNG，JPG，GIF和TIF。
- 在读取图片之前必须先创建窗口（因为创建图片时需要创建和窗口兼容的设备上下文句柄）。

第三节 读取图片

- 实例：

```
uses Display;//使用Display单元库
var img:pbitmap;
begin
CreateWin();//建立窗口
img:=LoadBMP('display.png');
Msgbox(i2s(img^.Width)+' '+i2s(img^.Height));//输出图片信息
end.
```

第四节 绘制图片

- 以下过程可以绘制图片：

```
procedure DrawBMP(b:pbitmap;xd,yd:longword);
procedure DrawBMP(bs,bd:pbitmap;xd,yd:longword);
```

- xd,yd为目标坐标，b,bs为需要绘制的图片，bd为绘制的目标。未指定bd时绘制到窗口。
- bar,line等绘图过程也可以绘制到图片，只需在第一个参数加入pbitmap的图片变量即可，具体参见Display单元库重载表。

第四节 绘制图片

- 实例：

```
uses Display;//使用Display单元库
var img1,img2:pbitmap;
begin
CreateWin(800,600);//建立窗口
img1:=LoadBMP('display.png');//读取图片
img2:=LoadBMP('display.png');//读取图片
DrawBMP(img1,(GetWidth()-img1^.Width)div
2,(GetHeight()-img1^.Height)div 2);//绘制图片1
```

第四节 绘制图片

```
FreshWin();//刷新窗口
Msgbox('图片1');//输出绘制完成信息
Bar(img2,img2^.Width div 4,img2^.Height div 4,img2^.Width div 2,img2^.Height div 2,Transparent,Blue);//绘制矩形到图片2中间
DrawBMP(img2,(GetWidth()-img2^.Width)div 2,(GetHeight()-img2^.Height)div 2);//绘制图片2
FreshWin();//刷新窗口
Msgbox('图片2');//输出绘制完成信息
end.
```

第五节 绘制拉伸图片

- 以下过程可以绘制拉伸的图片：

```
procedure DrawBMP(b:pbitmap;xd,yd,wd,hd:longword);
procedure
DrawBMP(b:pbitmap;xs,ys,ws,hs,xd,yd,wd,hd:longword);
```

- wd,hd为目标大小，不能为负数（不能反射）。
- xs,ys为需要绘制的图片左上角的位置，ws,hs为需要绘制的图片从xs,ys开始的大小。
- ws,hs必须比原始图片小，否则绘图将会失败。

第五节 绘制拉伸图片

- 实例：

```
uses Display;//使用Display单元库
var img:pbitmap;
begin
CreateWin(800,600);//建立窗口
img:=LoadBMP('display.png');//读取图片
DrawBMP(img,(GetWidth()-img^.Width*2)div
2,(GetHeight()-img^.Height*2)div
2,img^.Width*2,img^.Height*2);//绘制拉伸图片
```

第五节 绘制拉伸图片

```
FreshWin();//刷新窗口
Msgbox('绘制拉伸完成');//输出绘制完成信息
DrawBMP(img,img^.Width div 4,img^.Height div 4,img^.Width div 2,img^.Height div 2,(GetWidth()-img^.Width*2)div 2,(GetHeight()-img^.Height*2)div 2,img^.Width*2,img^.Height*2);//绘制剪切拉伸图片
FreshWin();//刷新窗口
Msgbox('绘制剪切完成');//输出绘制完成信息
end.
```

第六节 绘制透明图片

- 以下过程可以绘制透明和半透明的图片：

```
procedure DrawBMP(b:pbitmap;c:longword);
```

- c为透明颜色。如未指定c，则会使用b的背景颜色作为透明色。
- 该过程只会绘制图片中不是透明色的部分。
- 当c的Alpha部分不为0时（GetAlpha(c)<>0），绘图过程会以半透明的形式绘制到目标上（会创建临时位图并消耗资源，慎用）。

第六节 绘制透明图片

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin(800,600,Red);//建立窗口
DrawBMP(LoadBMP('display.png'),$7FFFFFFF);//绘制透明色半透明图片
FreshWin();//刷新窗口
Msgbox('绘制透明完成');//输出绘制完成信息
end.
```

第七节 快速画点

- 读取图片之前，请先创建pbitbuf类型变量。
- pbitbuf和bitbuf类型结构：

```
type pbitbuf=^bitbuf;
type bitbuf=record
bmi:BITMAPINFO;
len:longword;
buf:Pointer;
bmp:pbitmap;
end;
```

第七节 快速画点

- 一个位图缓存必须且只能和一个位图绑定。
- 请使用以下函数创建位图缓存：

```
function CreateBB(b:pbitmap):pbitbuf;
```

- 您可以使用以下函数直接获取窗口位图：

```
function GetWin():pbitmap;
```

- 绘制完毕后，请释放位图缓存：

```
procedure ReleaseBB(bb:pbitbuf);
```

第七节 快速画点

- 以下函数可以快速取点和画点：

```
function GetBBPixel(bb:pbitbuf;x,y:longword):longword;
procedure SetBBPixel(bb:pbitbuf;x,y,c:longword);
```

- 请使用以下过程绘制缓存到位图：

```
procedure SetBB(bb:pbitbuf);
```

- 您也可以读取位图到缓存：

```
procedure GetBB(bb:pbitbuf);
```

第七节 快速画点

- 实例：

```
uses Display;//使用Display单元库
var n:longword=$100000;//绘制点数量
var bb:pbitbuf;//位图缓存
begin
CreateWin(800,600,Red);//建立窗口
bb:=CreateBB(GetWin());//创建位图缓存
while n>0 do
```

第七节 快速画点

```
begin
SetBBPixel(bb,random(GetWidth()),random(GetHeight(
)),random($FFFFFF));//随机画点到缓存
n:=n-1;
end;
SetBB(bb);//绘制缓存到位图
FreshWin();//刷新窗口
Msgbox('快速画点完成');//输出绘制完成信息
end.
```

第四章 文字

- 第一节 输出文字
- 第二节 设定字体大小
- 第三节 设定字体

第一节 输出文字

- 以下过程可以输出文字：

```
procedure DrawText(s:ansistring;cfg,cbg:longword);
procedure DrawText(s:ansistring;c:longword);
procedure DrawText(s:ansistring);
```

- s为需要输出的字符串。
- cfg,cbg分别为文字的颜色和背景色。如果只指定c，则背景色为透明（不绘制背景色）。不指定c时，默认的文字颜色是窗体的前景色。

第一节 输出文字

- 可以使用XY系列过程将文字输出到指定位置：

procedure DrawTextXY(s:ansistring;x,y:longword);

- 或者ln系列过程按行效果输出：

procedure DrawTextln(s:ansistring);

- 使用w系列过程可输出定宽文本：

procedure DrawTextw(s:ansistring);

- 定宽字符宽度取决于字体，请先定义字体大小。
- 部分过程也可将文字输出到指定图片，具体请参见Display单元库重载表。

第一节 输出文字

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
DrawTextln('ax_pokl output text.');//输出文本并换行
DrawTextw('ax_pokl''s text is tight');//输出定宽文本（宽度默认为0）
DrawTextXY('ax_pokl output text anywhere',50,50);//指定位置输出文本
```

第一节 输出文字

```
DrawTextXY('',0,80);//强制改变输出位置
DrawText('and it''s colorful ',Orange);//输出带颜色文本
DrawText('with backgroud color',Red,Blue);//输出带背景颜色文本
FreshWin();//刷新窗口
Msgbox('绘制完成');//输出绘制完成信息
end.
```

第二节 设定字体大小

- 以下过程可以设定字体大小：

```
procedure SetFontWidth(w:longword);
procedure SetFontHeight(h:longword);
procedure SetFontSize(w,h:longword);
```

- w,h为宽和高，设为0时有特殊含义：
- h为0时，将使用系统默认的高度。
- w为0时，宽度将匹配高度。
- 设为0时，虽然显示的字体有宽和高，但其宽或高仍旧为0。

第二节 设定字体大小

- 实例：

```
uses Display;//使用Display单元库
var pyi:longword;
begin
CreateWin();//建立窗口
SetFontSize(5,10);//宽5,高10
DrawTextXY('5,10',0,0,White,Red);
SetFontHeight(20);//高20,宽不变
DrawTextXY('-,20',0,20,White,Red);
```

第二节 设定字体大小

```
SetFontSize(0,0);//默认大小
DrawTextXY('匹配,默认',0,40,White,Red);
SetFontSize(0,20);//高20,宽匹配
DrawTextXY('匹配,20',0,60,White,Red);
for pyi:=0 to 4 do
line(0,pyi*20,longint(GetWidth),0,Orange);
FreshWin();//刷新窗口
Msgbox('绘制完成');//输出绘制完成信息
end.
```

第三节 设定字体

- 以下过程可以设定字体：

```
procedure SetFontWeight(wg:longword);
procedure SetFontLtalic(lt:longword);
procedure SetFontUnderLine(ud:longword);
procedure SetFontStrikeOut(sk:longword);
procedure SetFontName(s:ansistring);
```

- 粗细wg默认为0，标准为400，粗体为700。
- 斜体lt，下划线ud，删除线sk为0或1。
- 用SetFont(b:pbitmap)可将当前字体选入图片。

第三节 设定字体

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
SetFontName('Comic Sans MS');//字体名称
DrawTextXY('Comic Sans MS',0,0);
SetFontWeight(700);//粗体
DrawTextXY('Weight',0,20);SetFontWeight(0);
```

第三节 设定字体

```
SetFontLtalic(1);//斜体
DrawTextXY('Ltalic',0,40);SetFontLtalic(0);
SetFontUnderLine(1);//下划线
DrawTextXY('UnterLine',0,60);SetFontUnderLine(0);
SetFontStrikeOut(1);//删除线
DrawTextXY('StrikeOut',0,80);SetFontStrikeOut(0);
FreshWin();//刷新窗口
Msgbox('绘制完成');//输出绘制完成信息
end.
```

第五章 消息

- 第一节 获取消息
- 第二节 处理消息
- 第三节 获取时间
- 第四节 获取帧率
- 第五节 控制帧率

第一节 获取消息

- 以下函数用以判断或获取消息：

function IsNextMsg():boolean;

function GetNextMsg():longword;

function WaitNextMsg():longword;

- 窗口线程中的消息会由线程自动发送给窗口，然后同时自动刷新消息缓存。
- 对于非线程消息，窗口线程不会发送。
- IsNextMsg会返回队列中是否有新消息。GetNextMsg和WaitNexgMsg会返回消息号。

第一节 获取消息

- 在消息缓存中被放入新消息之前，WaitNextMsg不会返回。IsNextMsg和GetNextMsg会立即返回，无论是否有新消息。
- 使用GetNextMsg前请务必先使用IsNextMsg或WaitNextMsg刷新窗口消息缓存计数，否则将取不到下一条消息。
- GetNextMsg会返总是返回刷新消息缓存计数后的当前计数的消息的消息号。

第一节 获取消息

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
repeat//第一种消息循环
if IsNextMsg() then SetTitle(i2s(GetNextMsg()))//如果有新消息则输出消息号到标题栏
else Delay();//否则等待1毫秒
until not(IsWin()) or (IsKey(27));//直到关闭窗口或按ESC
```

第一节 获取消息

```
while IsWin() do//如果窗口开着则循环
begin//第二种消息循环
WaitNextMsg();//等待新消息
SetTitle(i2s(GetNextMsg()));//输出消息号到标题栏
if IsKey(27) then CloseWin();//如果是按ESC则关闭窗口
end;//直到关闭窗口
end.
```

第二节 处理消息

- 获取消息后，可以使用以下函数进行处理

function IsMsg(uM:longword):boolean;

function GetMsg(uM:longword):qword;

function WaitMsg(uM:longword):qword;

- IsMsg用以判断当前消息是否指定消息。
- GetMsg可获取消息的参数。如果当前消息不是指定消息，则函数返回0。
- WaitMsg会等待指定消息并返回消息的参数。

第二节 处理消息

- 以下函数可以判断特定类型消息：

```
function IsKey():boolean;
function IsKey(k:longword):boolean;
function IsMouse():boolean;
function IsMouse(m:longword):boolean;
function IsMouseLeft():boolean;
function IsMouseMove():boolean;
function IsDropFile():boolean;
```

- 部分以上函数也有Get和Wait的版本。

第二节 处理消息

- 以下函数可获取鼠标的位置：

```
function GetMouseAbsX():longint;
function GetMouseAbsY():longint;
function GetMouseWinX():longint;
function GetMouseWinY():longint;
function GetMousePosX():longint;
function GetMousePosY():longint;
```

- Abs，Win和Pos分别为鼠标的绝对坐标，窗口坐标和绘图区坐标。

第二节 处理消息

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
repeat
SetTitle(i2s(GetMousePosX())+'
'+i2s(GetMousePosY()));//输出鼠标位置到标题栏
WaitNextMsg();
if IsKey() then MsgBox(i2s(GetKey()));//如果是按键则输出按键号
```

第二节 处理消息

if IsMouse() then Msgbox(i2s(GetMouse()));//如果按鼠标则输出鼠标号

if IsMouseWheel() then Msgbox(i2s(GetMouseWheel()));//如果鼠标滚轮则输出滚轮号

if IsDropFile() then Msgbox(GetDropFile());//如果是拖拽文件则输出文件名

until not(IsWin()) or (IsKey(27));//直到关闭窗口或按ESC键

end.

第三节 获取时间

- 以下函数可以获取时间：

function GetTimeR():real;

function GetTime():longword;

- GetTimeR返回从窗口建立开始到现在的时间。
- GetTime返回整型时间，以毫秒计。

第三节 获取时间

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
repeat
SetTitle(i2s(GetTime()));//输出时间到标题栏
IsNextMsg();Delay();////等待新消息并延迟1毫秒
until not(IsWin()) or IsKey();//直到窗口关闭或按键
end.
```

第四节 获取帧率

- 以下函数可以获取帧率：

function GetFPSL():longword;

function GetFPSR():real;

function GetFPS():longword;

- GetFPSL返回从一秒前开始到当前的帧数（刷新次数，即调用FreshWin的次数）。
- GetFPSR返回GetFPSL*一秒前开始第一帧到当前帧（最后刷新）的时间（时间小于1秒）。
- GetFPS返回GetFPSR取整的结果。

第四节 获取帧率

- 实例：

```
uses Display;//使用Display单元库
begin
CreateWin();//建立窗口
repeat
IsNextMsg();//等待新消息
Clear();//清屏
DrawTextXY('',0,0);//设置文本输出位置
```

第四节 获取帧率

```
DrawTextln(i2s(GetFPSL()));//输出瞬时刷新率
DrawTextln(i2s(GetFPS()));//输出平均刷新率
FreshWin();//刷新窗口
Delay();//延迟1毫秒
until not(IsWin()) or IsKey();//直到窗口关闭或按键
end.
```

第五节 处理帧率

- 以下过程可以延迟时间：

procedure Delay(t:longword);

procedure Delay();

- t为longword时以毫秒计。
- 最短延迟时间视系统状态而定，这可能是1000/60毫秒或者1毫秒（1系统tick）。

第五节 处理帧率

- 实例：

```
uses Display;//使用Display单元库
var frame:real=120.0;//帧率
var frametime:real=0.0;//当前帧时间
begin
CreateWin();//建立窗口
repeat
if IsNextMsg() then//如果有新消息
begin
```

第五节 处理帧率

if (frame>10) and IsKey(37) then frame:=frame-1;//如果按左则减小帧率

if (frame<480) and IsKey(39) then frame:=frame+1;//如果按右则增加帧率

end;

if GetTimeR()>frametime+1/frame then//如果当前时间已超过一帧时间

begin

while GetTimeR()>frametime+1/frame do frametime:=frametime+1/frame;//增加帧数（包括跳帧）

第五节 处理帧率

```
Clear();//清屏
DrawTextlnXY(i2s(GetFPSL()),0,0);//输出瞬时刷新率
DrawTextln(i2s(GetFPS()));//输出平均刷新率
DrawTextln(i2s(longint(round(frame))));//输出瞬时刷新率
FreshWin();//刷新窗口
end;
Delay();//延迟1毫秒
until not(IsWin()) or IsKey(27);//直到窗口关闭或按ESC
end.
```

第六章 音频

- 第一节 读取音频
- 第二节 播放音频
- 第三节 设定音量
- 第四节 暂停音频
- 第五节 跳转音频
- 第六节 音频播放器

第一节 读取音频

- 以下函数可以读取音频，请在读取之前先创建longword类型变量：

function LoadAudio(s:ansistring):longword;

- 之后对音频的操作需要这个longword类型变量。
- 可以同时读取多个音频，用longword变量区分。
- 支持的音频格式有wav，mp3，wmv等系统内生支持的格式，和Windows media player相同。

第一节 读取音频

- 实例：

```
uses Display;//使用Display单元库
var audio:longword;
begin
audio:=LoadAudio('display.mp3');//读取音频
Msgbox(i2s(audio));//输出音频号
end.
```

第二节 播放音频

- 以下过程可以播放音频：

```
procedure PlayAudio(id:longword;b:boolean);
procedure PlayAudio(id:longword);
```

- b为true时，音频将重复播放。
- 不指定b时默认为false（单曲播放）。
- 实例：

```
uses Display;//使用Display单元库
var audio1,audio2:longword;
```

第二节 播放音频

```
begin
audio1:=LoadAudio('display.mp3');//读取音频1
audio2:=LoadAudio('display.mp3');//读取音频2
PlayAudio(audio1);//播放音频1
Msgbox('正在播放音频');
PlayAudio(audio2,true);//重复播放音频2
Msgbox('正在重复播放音频');
end.
```

第三节 设定音量

- 播放过程中可获取或设定音频音量：

function GetAudioVol(id:longword):longword;

procedure SetAudioVol(id:longword;v:longword);

- 音量v为longword类型，范围为0到1000。
- 每个音频可以设定不同的音量。
- 音频必须在开始播放以后才能设定音量。

第三节 设定音量

- 实例：

```
uses Display;//使用Display单元库
var audio:longword;
begin
audio:=LoadAudio('display.mp3');//读取音频
PlayAudio(audio);//播放音频
SetAudioVol(audio,200);//设定音量
Msgbox('正在播放音频，音量200');
end.
```

第四节 暂停音频

- 以下过程可以实现音频的暂停，继续，停止：

```
procedure PauseAudio(id:longword);
procedure ResumeAudio(id:longword);
procedure StopAudio(id:longword);
procedure ReleaseAudio(id:longword);
```

- Pause和Resume可以暂停，继续音频的播放。
- 用Stop停止播放音频后，可以用Play重新播放。
- 如需彻底将音频从内存中释放，请用Release。

第四节 暂停音频

- 实例：

```
uses Display;//使用Display单元库
var audio:longword;
begin
audio:=LoadAudio('display.mp3');//读取音频
PlayAudio(audio);//播放音频
Msgbox('正在播放音频，按确定暂停');
PauseAudio(audio);
Msgbox('音频已暂停，按确定继续');
```

第四节 暂停音频

```
ResumeAudio(audio);
Msgbox('音频已继续重复播放');
StopAudio(audio);
Msgbox('音频已停止播放');
PlayAudio(audio);
Msgbox('音频已重新开始播放');
ReleaseAudio(audio);
Msgbox('音频已释放');
end.
```

第五节 跳转音频

- 以下过程可以实现音频的跳转，以及获取音频的播放位置和长度：

function GetAudioLen(id:longword):longword;

function GetAudioPos(id:longword):longword;

procedure SetAudioPos(id:longword;pos:longword);

procedure SetAudioPos(id:longword;pos:longword;b:boolean);

- 获取的位置和长度为longword，以毫秒计。
- 如需从指定位置重复播放，请设b为true。

第五节 跳转音频

- 实例：

```
uses Display;//使用Display单元库
var audio:longword;
begin
audio:=LoadAudio('display.mp3');//读取音频
PlayAudio(audio);SetAudioPos(audio,GetAudioLen(audio)div 2);//从中间开始播放音频
Msgbox('正在播放音频，从中间开始');
end.
```

第七章 应用

- 第一节 音频播放器
- 第二节 俄罗斯方块

第一节 音频播放器

- 使用前六章知识，编写简易音频播放器：

```
uses Display;//使用Display单元库
var w:longword=600;//窗口宽
var h:longword=100;//窗口高
var frame:real=120.0;//帧率
var frametime:real=0.0;//当前帧时间
var audio:longword;//音频句柄
var pos:longword;//音频窗口位置
var play:boolean=false;//音频播放状态
```

第一节 音频播放器

```
begin
CreateWin(w,h,blue);//建立蓝色窗口
SetTitle('display.mp3');//设定标题
audio:=LoadAudio('display.mp3');//读取音频
PlayAudio(audio);//播放音频
repeat//开始消息循环
if IsNextMsg() then//如果有新消息
begin
if IsDropFile() then//如果有拖拽文件
```

第一节 音频播放器

```
begin
SetTitle(GetDropFile());//设定标题为拖拽文件名
StopAudio(audio);//停止正在播放的音频
audio:=LoadAudio(GetDropFile());//读取音频
PlayAudio(audio);//播放音频
play:=true;//设定音频状态
end;
if IsKey(37) then//如果按左
SetAudioPos(audio,max(GetAudioPos(audio)-
1000,0));//倒退1秒
```

第一节 音频播放器

if IsKey(39) then//如果按右

SetAudioPos(audio,min(GetAudioPos(audio)+1000,GetAudioLen(audio)));//前进1秒

if IsKey(40) then//如果按下

SetAudioVol(audio,max(GetAudioVol(audio)-100,0));//减小100音量

if IsKey(38) then//如果按上

SetAudioVol(audio,min(GetAudioVol(audio)+100,1000));//增大100音量

if IsMouseLeft() then//如果鼠标左键

第一节 音频播放器

```
SetAudioPos(audio,round(GetMousePosX()/real(w)*GetAudioLen(audio)));//跳转音频
if IsMouseRight() or IsKey(32) then//如果鼠标右键或按空格
begin
if play then PauseAudio(audio)//如果正在播放则暂停
else ResumeAudio(audio);//否则继续播放
play:=not(play);//更改音频状态
end;
end;
```

第一节 音频播放器

if GetAudioPos(audio)=GetAudioLen(audio) then//如果已播放完毕

SetAudioPos(audio,0);//重头播放

if GetTimeR()>frametime+1/frame then//如果当前时间已超过一帧时间

begin

while GetTimeR()>frametime+1/frame do frametime:=frametime+1/frame;//增加帧数（包括跳帧）

if GetAudioLen(audio)=0 then pos:=0//如果音频长度为0（没有音频）则设音频窗口位置为0

第一节 音频播放器

```
else pos:=round(real(w)*GetAudioPos(audio)/
GetAudioLen(audio));//否则设定音频窗口位置
Clear();Bar(pbitmap(0),0,0,pos,100,Transparent,Yellow
);//绘制状态
DrawTextInXY(i2s(GetAudioPos(audio))+' /
'+i2s(GetAudioLen(audio)),0,0,Yellow,Blue);//输出状态
FreshWin();end;//刷新窗口
Delay();//延迟1毫秒
until not(IsWin()) or (IsKey(27));//直到关闭窗口或按ESC
end.
```

第二节 俄罗斯方块

- 一款简易的俄罗斯方块小游戏：

```
uses Display;//使用Display单元库
const w=10;//场地宽
const h=20;//场地高
var sz:longword=30;//方块大小
var frame:real=120.0;//帧率
var frametime:real=0.0;//当前帧时间
var downtime:real=0.0;//下落时间
```

第二节 俄罗斯方块

```
var i,j:shortint;//场地行列计数
var x,y,r,k:shortint;//当前方块状态
var bd:array[0..w-1,0..h-1]of shortint;//场地方块
const bdc:array[0..7]of longword=
($1F1F1F,$7F7F7F,$7F7FFF,$7FFF7F,$FF7F7F,$7FFFFF,$FFFF7F,$FF7FFF);//方块颜色
const bdk:array[0..7,0..3,0..3,0..3]of longword;//方块类型（已省略）
```

第二节 俄罗斯方块

```
procedure DrawBlock(i,j,k:shortint);//画方块
begin Bar(i*sz,(h-j-1)*sz,sz,sz,bdc[k]);end;
procedure NewBlock();forward;//新方块
procedure Restart();//重新开始
begin
for i:=0 to w-1 do for j:=0 to h-1 do bd[i,j]:=0;//清空场地
NewBlock();//新方块
end;
```

第二节 俄罗斯方块

```
function EraseLine():boolean;//消行
begin
for j:=0 to h-1 do//从最底行开始
begin
EraseLine:=true;//是满行
for i:=0 to w-1 do if bd[i,j]=0 then EraseLine:=false;//如果有洞则不是
if EraseLine then break;//如果是满行则跳出
end;
```

第二节 俄罗斯方块

```
if EraseLine then//如果是满行（消行）
while j<(h-1) do//从此行开始（往上）
begin
for i:=0 to w-1 do//遍历该行
bd[i,j]:=bd[i,j+1];//上方方块掉落
j:=j+1;//继续上一行
end;
end;
```

第二节 俄罗斯方块

```
procedure FixBlock();//固定方块（落底）
begin
for i:=0 to 3 do for j:=0 to 3 do//遍历方块行列
if bdk[k,r,j,i]>0 then bd[i+x,j+y]:=k;NewBlock();//如果是格子非空则画到场地
while EraseLine() do ;//消行
end;
```

第二节 俄罗斯方块

```
function Overlay():boolean;//判断重叠
begin
Overlay:=false;//设非重叠
for i:=0 to 3 do for j:=0 to 3 do//遍历方块行列
if (bdk[k,r,j,i]>0) then//如果格子非空
if (i+x<0)or(i+x>=w)or(j+y<0)or(j+y>=h) then
Overlay:=true//如果超出场地则重叠
else if (bd[i+x,j+y]>0) then Overlay:=true;//如果没超出场地但场地非空也重叠
end;
```

第二节 俄罗斯方块

```
procedure NewBlock();//新方块
begin
x:=3;//新方块行
y:=16;//新方块列
r:=0;//新方块旋转
k:=random(7)+1;//新方块类型
if Overlay() then Restart();//如果重叠则重来
end;
```

第二节 俄罗斯方块

```
function Rotate(d:shortint):boolean;//旋转
begin
r:=r+1;if r>3 then r:=0;Rotate:=not(Overlay());//尝试旋转
if not(Rotate) then r:=r-1;if r<0 then r:=3;//如果不能旋转则转回来
end;
```

第二节 俄罗斯方块

```
function Move(dx,dy:shortint):boolean;//移动
begin
x:=x+dx;y:=y+dy;Move:=not(Overlay());//尝试移动
if not(Move) then begin x:=x-dx;y:=y-dy;end;//如果不能移动则移回来
if not(Move) and (dy<0) then FixBlock();//如果不能移动且下落则固定
if dy<0 then downtime:=GetTimeR();//如果下落则重置下落时间
end;
```

第二节 俄罗斯方块

```
begin//主程序
Randomize();//初始化随机种子
CreateWin(w*sz,h*sz);//建立窗口
SetTitle('俄罗斯方块');//设定标题
Restart();//重新开始
repeat//开始消息循环
if IsNextMsg() then//如果有新消息
begin
```

第二节 俄罗斯方块

```
if IsKey(37) then Move(-1,0);//如果按左则左移
if IsKey(39) then Move(+1,0);//如果按右则右移
if IsKey(40) then Move(0,-1);//如果按下则下落
if IsKey(38) then Rotate(1);//如果按上则旋转
if IsKey(32) then while Move(0,-1) do ;//如果按空格则下底
end;
if GetTimeR()>downtime+1 then Move(0,-1);//如果超过1秒则下落
```

第二节 俄罗斯方块

```
if GetTimeR()>frametime+1/frame then//如果当前时间已超过一帧时间
begin
while GetTimeR()>frametime+1/frame do frametime:=frametime+1/frame;//增加帧数（包括跳帧）
Clear();
for i:=0 to w-1 do for j:=0 to h-1 do DrawBlock(i,j,bd[i,j]);//画场地
for i:=0 to 3 do for j:=0 to 3 do if bdk[k,r,j,i]>0 then DrawBlock(i+x,j+y,k);//画当前方块
```

第二节 俄罗斯方块

```
FreshWin();//刷新窗口
end;
Delay();//延迟1毫秒
until not(IsWin()) or (IsKey(27));//直到关闭窗口或按ESC键
end.
```

附录

- Display单元库重载表

Display单元库重载表

- 详情请参阅display.pp，以下列出部分重载：

```
function i2s(i:longword):ansistring;
function NewThread(th:pointer):longword;
procedure MsgBox(s,title:ansistring);
procedure MsgBox(s:ansistring);
procedure Delay(t:longword);
function GetFPSR():real;
function GetFPS():longword;
function GetError():longword;
```

Display单元库重载表

```
procedure CreateWin(w,h:longword;cfg,cbg:longword);
procedure CreateWin(w,h:longword;c:longword);
procedure CreateWin(w,h:longword);
procedure CreateWin(c:longword);
procedure CreateWin();
procedure FreshWin();
procedure CloseWin();
function IsWin():boolean;
procedure SetDrawProcedure(th:tprocedure);
```

Display单元库重载表

```
procedure SetTitle(s:ansistring);
function GetTitle():ansistring;
function GetTimeR():real;
function GetTime():longword;
function GetWidth():longword;
function GetHeight():longword;
function GetSize():longword;
function GetPosX():longint;
function GetPosY():longint;
```

Display单元库重载表

```
function GetBGColor():longword;
procedure SetBGColor(c:longword);
function GetFGColor():longword;
procedure SetFGColor(c:longword);
function GetBlue(c:longword):byte;
function GetGreen(c:longword):byte;
function GetRed(c:longword):byte;
function GetAlpha(c:longword):byte;
function GetRGB(r,g,b:byte):longword;
```

Display单元库重载表

procedure SetFont(b:pbitmap);
procedure SetFontWidth(w:longword);
procedure SetFontHeight(h:longword);
procedure SetFontSize(w,h:longword);
procedure SetFontWeight(wg:longword);
procedure SetFontLtalic(lt:longword);
procedure SetFontUnderLine(ud:longword);
procedure SetFontStrikeOut(sk:longword);
procedure SetFontName(s:ansistring);

Display单元库重载表

procedure DrawTextXY(b:pbitmap;s:ansistring;
x,y:longword;cfg,cbg:longword);
procedure DrawTextXY(s:ansistring;
x,y:longword;cfg,cbg:longword);
procedure DrawTextXY(s:ansistring;
x,y:longword;c:longword);
procedure DrawTextXY(s:ansistring;x,y:longword);
procedure DrawText(s:ansistring);
procedure DrawTextln(s:ansistring);

Display单元库重载表

```
function GetPixel(x,y:longword):longword;
procedure SetPixel(x,y:longword;c:longword);
procedure Line(x,y:longword;w,h:longint;c:longword);
procedure Bar(x,y:longword;w,h:longint;
cfg,cbg:longword);
procedure Bar(x,y:longword;w,h:longint;c:longword);
procedure Clear(c:longword);
procedure Circle(x,y,r:longint;c:longword);
procedure Ellipse(x,y,rx,ry:longint;c:longword);
```

Display单元库重载表

```
function CreateBMP(w,h:longword):pbitmap;
function LoadBMP(s:ansistring):pbitmap;
procedure ReleaseBMP(b:pbitmap);
procedure DrawBMP(bs,bd:pbitmap;
xs,ys,ws,hs,xd,yd,wd,hd:longword;c:longword);
procedure DrawBMP(b:pbitmap;
xs,ys,ws,hs,xd,yd,wd,hd:longword;c:longword);
procedure DrawBMP(b:pbitmap;
xs,ys,xd,yd,w,h:longword;c:longword);
```

Display单元库重载表

procedure DrawBMP(b:pbitmap;
xd,yd,wd,hd:longword;c:longword);
procedure DrawBMP(b:pbitmap;
xd,yd:longword;c:longword);
procedure DrawBMP(b:pbitmap;c:longword);
procedure DrawBMP(b:pbitmap);
procedure DrawBMP(xd,yd,wd,hd:longword);
procedure DrawBMP(xd,yd:longword);
procedure DrawBMP();

Display单元库重载表

function IsNextMsg():boolean;
function GetNextMsg():longword;
function WaitNextMsg():longword;
function IsKey():boolean;
function IsKey(k:longword):boolean;
function IsMouse():boolean;
function IsMouse(m:longword):boolean;
function IsMouseLeft():boolean;
function IsMouseMiddle():boolean;

Display单元库重载表

function IsMouseRight():boolean;
function IsMouseWheel():boolean;
function GetMouseWheel():longint;
function IsMouseMove():boolean;
function GetMouseMove():longword;
function IsDropFile():boolean;
function GetDropFile():ansistring;
function GetMousePosX():longint;
function GetMousePosY():longint;

Display单元库重载表

function LoadAudio(s:ansistring):longword;
procedure PlayAudio(id:longword;s:ansistring;
b:boolean);
procedure PlayAudio(id:longword;s:ansistring);
procedure PlayAudio(id:longword;b:boolean);
procedure PlayAudio(id:longword);
procedure PauseAudio(id:longword);
procedure ResumeAudio(id:longword);

Display单元库重载表

procedure StopAudio(id:longword);
procedure ReleaseAudio(id:longword);
function GetAudioVol(id:longword):longword;
procedure SetAudioVol(id:longword;v:longword);
function GetAudioLen(id:longword):longword;
function GetAudioPos(id:longword):longword;
procedure SetAudioPos(id:longword;pos:longword);
procedure SetAudioPos(id:longword;pos:longword;
b:boolean);

Display单元库重载表

- Display单元库还有字符串处理子程序和大数处理（利用字符串，速度慢）的子程序。
- Display单元库也支持基本的文件块状读取和文件、文件夹的操作。
- 此外，使用Display单元库还可以模拟简单的鼠标和键盘的操作（应用级别）。
- 以上有关的子程序就不做介绍了，有兴趣的同学可以自己阅读Display单元库的源码。

后记

- 使用简单的办法用Pasca语言开发窗体应用软件和游戏是我的愿望，因此我便编写了Display单元库。现在，这个愿望已经实现了。
- 随着智能手机的普及，我希望Display单元库不仅局限于Windows操作系统，使得它未来能在各种Linux，Mac OS和Andriod系统上运行。
- 从开始只有几十个子程序，经过五年的修改，Display单元库已知在不断的完善和扩充。对于库的错误和建议，也请各位大神多多指点。