

**ErrorSoft Low Graphics Library**

**(ESLGL)**

This lib use in my demo “n.29”:

<http://www.youtube.com/watch?v=zL_2NWtxLQE&feature=youtu.be>

<http://www.pouet.net/prod.php?which=64031>

***Many Sample***

#include "Graphics.h"

#define clBlack 0

#define clWhite 1

#define clInvert 2

main(void)

{

PBitMap BitMap;

BitMap = esCreateBitMap(100, 100, pf1bit);

esClear(BitMap, clWhite);// fill canvas

esDrawRect(BitMap, 0, 0, 99, 99, clBlack);

esFillElipse(BitMap, 0, 0, 99, 99, clBlack);

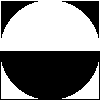
esFillRect(BitMap, 0, 0, 99, 49, clInvert);

YourSystemDrawToScreenFunction(BitMap);// <- Write you!

esFreeBitMap(BitMap);

return 0;

}



**It’s Simple!**

**Overview**

ESLGL – this is low-end graphics library, optimized to 4-bit and 1-bit per pixel graphics.

Library written on “C”, but imitate PLO.

Library supports these operations:

* BitMap`s
  + esCreateBitMap – Create new BitMap
  + esCreateStaticBitMap – Create new BitMap using a static map pixels
  + esCreateStaticMaskBitMap – Create new masked BitMap using a static map pixels
  + esResizeBitMap – Resize BitMap
  + esCloneBitMap – Clone this BitMap
  + esFreeBitMap – Free BitMap
  + esFreeStaticBitMap – Free static BitMap
* BitMap effects
  + esCopyBitMap
  + esInverseBitMap
  + esFlipVBitMap
  + esFlipHBitMap
  + esRotate90BitMap
* Main graphics
  + esClear
  + esSetPixel
  + esGetPixel
  + esFillRect
  + esDrawRect
  + esDrawLine
  + esDrawElipse
  + esFillElipse
* The BitBlt functions
  + **1 - bit**
    - esBitBlt01\_Copy
    - …
    - esBitBlt01\_Mask
    - esBitBlt01
    - esBitBltRop01
  + **4 – bit**
    - esBitBlt04\_Copy
    - …
    - esBitBlt04\_Color
    - esBitBltRop04
    - esBitBlt04
  + **Universal**
    - esBitBlt
    - esBitBltRop
* Strech draw functions
  + **1 – bit**
    - esStrechDraw01\_Copy
    - esStrechDraw01\_Or
    - esStrechDraw01\_Xor
    - esStrechDraw01\_And
    - esStrechDraw01\_Mask
    - -
    - esStrechDraw01
    - esStrechDrawRop01
  + **4 – bit**
    - esStrechDraw04\_Copy
    - esStrechDraw04\_Or
    - esStrechDraw04\_Xor
    - esStrechDraw04\_And
    - esStrechDraw04\_Mask
    - esStrechDraw04\_Transparent
    - esStrechDraw04\_Color
    - -
    - esStrechDraw04
    - esStrechDrawRop04
  + **Universal**
    - esStrechDraw
    - esStrechDrawRop
* other …….

**Structurs**

**TPoint**

Simple 2d point.

typedef struct

{

int x, y;

} TPoint, \*PPoint;

**TRect**

Recangle.

typedef struct

{

int x1, y1;

int x2, y2;

} TRect, \*PRect;

**TColor**

typedef unsigned TColor;

**TPixelFormat**

typedef enum

{

pf1bit = 1,

pf2bit = 2,

pf4bit = 4,

pfC2 = 8

} TPixelFormat;

**TAlphaFormat**

typedef enum

{

afNone = 0,

afMask = 16,

afColor = 32

} TAlphaFormat;

**TRop**

This operation apply for BitBlt functions.

typedef enum

{

ropCopy,

ropOr,

ropXor,

ropAnd,

ropNotAnd

} TRop;

**TBitMap**

Main struct library.

typedef struct

{

int Width;

int Height;

unsigned char \*Pixels;

unsigned char \*Mask;

TPixelFormat PixelFormat;

TAlphaFormat AlphaFormat;

TColor TransparentColor;//todo: add default

// precalc

int Scanline;

} TBitMap, \*PBitMap;

Pixels – pixels map

Mask – pixels map for mask

Scanline – precalc value, for height performance

* Use if define USE\_PRECALC\_SCANLINE

**In the process of writing, I will be happy to help with documentation…**