Subject: Draw stuff [SOLVED]
Posted by Indio on Fri, 01 Feb 2008 04:21:36 GMT
View Forum Message <> Reply to Message

Hi all!

I'm newbie with ultimate++ and I have got a little problem with drawing.

I'd like to draw a simple image on the GUI. Where's the error in the following source?

```
  //------------------------------------------------------------ --

ImageBuffer ib( 128, 128 );

for( int y = 0; y < 128; y++) {
        RGBA *l = ib[ y ];
        for( int x = 0; x < 128; x++) {

            // in picture[ ][ ] there is the image
            l->a = 255;
            l->r = picture[ y ][ x ];
            l->g = picture[ y ][ x ];
            l->b = picture[ y ][ x ];
            l++;
       }
}

// Premultiply ( ib );  according to the example, this line is needed, but the compiler gives an error
-> the example is not perfect

Image proba_image = ib;
I guess so far the stuff is good

  //------------------------------------------------------------ -----------

void MyDraw (  )
{
        ImageDraw idraw( 128, 128 );
        s.Paint( idraw );    //  s is one of the Splittes, I divided the main window
}

  //------------------------------------------------------------ ----------

void Paint ( Draw& w )
{
    if ( imageLoaded ) // this is true, when the picture is loaded into the memory ( this happens )
    {
         Image img = proba_image;   // this works
```

```
        w.DrawRect ( 10, 10, 128, 128, White );
        w.DrawImage ( 10, 5, img );
    }

    // Refresh(  ); the stuff dies, when this line is active

}

  //---------------------------------------------------------- -----------
```

MyDraw() is called in a proper place, after the image is loaded. But, as a result, nothing happens, the image doesn't apppears on the GUI.

Can you help me in this problem? Thanks in advance!

Subject: Re: Draw stuff
Posted by mrjt on Fri, 01 Feb 2008 10:19:45 GMT
View Forum Message <> Reply to Message

Try this:

```
// Member variables
Image proba_image;

void LoadImage()
{
  ImageBuffer ib( 128, 128 );

  for( int y = 0; y < 128; y++) {
    RGBA *l = ib[ y ];
    for( int x = 0; x < 128; x++) {
      // in picture[ ][ ] there is the image
      l->a = 255;
      l->r = picture[ y ][ x ];
      l->g = picture[ y ][ x ];
      l->b = picture[ y ][ x ];
      l++;
    }
  }

  Premultiply ( ib );
  proba_image = ib;
}
```

```
void Paint ( Draw& w )
{
  w.DrawRect ( GetSize(), White );
  w.DrawImage ( 10, 5, proba_image );
}
```
- I can't think of any reason for MyDraw, what are you trying to achieve?
- The original problem may have been declaring proba_image locally, not as a memeber variable, so that it went out of scope.
- You don't need the imageLoaded flag. It is perfectly safe to draw an empty image, or you can use Image::IsEmpty();
- You probably don't need the picture[] array. You could either read whatever data it is directly into an ImageBuffer or if it is an image file load it directly.
- Premultiply is probably necessary. If the compiler complains then your version of uppsrc may be older than the example (though I'm not sure how that could happen).

Edit: Also it would be helpful if you could use a more descriptve title for new threads. Everything in this topic is 'Draw Stuff'

Hope that helps.
James

Subject: Re: Draw stuff
Posted by Indio on Fri, 01 Feb 2008 15:09:48 GMT
View Forum Message <> Reply to Message

James!

You're right about the title! Sry I can't rename it. If a moderator comes around, he can rename it e.g. "Iterative drawing problem". You will see why.

My program will segments images with energy minimization, if I ever finish it. The algorithm iterates, and it produces a 2D matrix (an image) in every iteration, and I want to draw this matrix onto the GUI and refresh it, if the matrix has changed.

But first of all I would like to put a simple image onto the GUI from the memory. If I can do this, I will try the advanced version mentioned above.

You're code has the same result as mine... Nothing appeared.

- The meaning of MyDraw was that I called it right after the image was loaded to the memory by the user. And if I don't call the Paint method on my object (here is the Splitter, named 'a') in MyDraw, how will it know where to paint exactly? My main window is divided by 3 Splitters. Or the w.DrawImage(x, y, image) draws simply on the main window? I've got no derived widgets.

- There's no problem with proba_image. It's not empty and it is a member of course. I wrote that this part of the code works. The problem comes up in painting.

- IsEmpty() is a good idea.

- I need **picture, becouse I modify the loaded image, and I that modifyed stuff has to be represented on the GUI.

- If I don't comment out Premultiply, the compiler complains that it is undeclared. I've got the 2007.1 version. I have been using it from November I guess.

I know that it is an easy problem, but I was shot. Any more idea? Thanks.

Subject: Re: Draw stuff
Posted by mrjt on Fri, 01 Feb 2008 15:44:08 GMT
View Forum Message <> Reply to Message

I see the problem now. You are trying to draw the image into a Splitter, but the Paint routine is a member of the main window yes? If that is correct then the image is being drawn and then the Splitter is drawn over the top.

What you need to do is add a Ctrl to the Splitter, and have the ctrl draw the Image. You can use ImageCtrl for this, though you may with to roll your own once you have it working.

You don't need the Paint function or MyDraw, just:

```
// Member variables
ImageCtrl imgctrl;

void LoadImage()
{
  ImageBuffer ib( 128, 128 );

  for( int y = 0; y < 128; y++) {
    RGBA *l = ib[ y ];
    for( int x = 0; x < 128; x++) {
      // in picture[ ][ ] there is the image
      l->a = 255;
      l->r = picture[ y ][ x ];
      l->g = picture[ y ][ x ];
      l->b = picture[ y ][ x ];
      l++;
    }
  }

  imagectrl.SetImage(ib);
}

// And in your window constuctor
..
```

splitter.Add(imgctrl);
..

PreMultiply is only needed for versions later the 2007.1, but if I were you I would upgrade to the latest dev version and add the call. Dev versions are always very stable and then you won't have to remember to do it later, plus it's been a long time since 2007.1 and there are manby improvements.

Subject: Re: Draw stuff
Posted by Indio on Fri, 01 Feb 2008 20:38:46 GMT
View Forum Message <> Reply to Message

I wanted to try out the way you mentioned, but there is something I don't undestand. If I define the ImageBuffer ib(height, width) locally in the LoadImage( ), then everything is fine. But If I do this as a member value, then "no match for call to `(Upp::ImageBuffer) (int&, int&)". height and width are also member values.

Subject: Re: Draw stuff
Posted by mirek on Fri, 01 Feb 2008 21:42:27 GMT
View Forum Message <> Reply to Message

Indio wrote on Fri, 01 February 2008 15:38I wanted to try out the way you mentioned, but there is something I don't undestand. If I define the ImageBuffer ib(height, width) locally in the LoadImage( ), then everything is fine. But If I do this as a member value, then "no match for call to `(Upp::ImageBuffer) (int&, int&)". height and width are also member values.

Hard to say without seeing the code.

Post it please.

Mirek

Subject: Re: Draw stuff
Posted by Indio on Fri, 01 Feb 2008 22:41:51 GMT
View Forum Message <> Reply to Message

That's it. I have done some selections, only the important parts of the source are here.

MainWindow.h :

```
#include "Segmentation.h"
#include <CtrlLib/CtrlLib.h>
```

```
class Segmentation; // forward class declaration

class MainWindow : public TopWindow {

public:
 typedef MainWindow CLASSNAME;
 MainWindow(  );

 void OpenImage(  );

 ImageCtrl inImgctrl;
 ImageCtrl outImgctrl;

 Splitter h, v, v1, v2;
       Button a;
 One < StreamRaster > r;

 Segmentation *segmentation;

};
```

MainWindow.cpp :

```
#include "MainWindow.h"

using namespace Upp;

//---------------------------

MainWindow::MainWindow(  )
{
 segmentation = NULL;
 segmentation = new Segmentation(  );

 Title("Statistical Segmentation").Zoomable().Sizeable();
 SetRect(0, 0, 900, 800);

 h.Horz(a, v);
 v.Vert(v1, v2 );
 Add(h.SizePos());
 v1.Add(inImgctrl);
}

//---------------------------
```

```
void MainWindow::OpenImage(  )
{
  String fileName = "";
  FileSel fs;

  // If user selects a file to open, returns true
  if ( fs.Type( "bitmap", "*.bmp").ExecuteOpen("Choose the image file to open") )
    {
      fileName = ~fs; // fileName contains the file name

      if ( fileName != "" )
      {
 inImgctrl.SetImage( Null ); // maybe not necessary
 FileIn in( fileName );
 r = StreamRaster::OpenAny( in );
 Image img = StreamRaster::LoadFileAny(~fileName);

        if ( !r )
        {
          return; // invalid input
 }

        ...

 if ( segmentation->LoadImage( r ) )
 {
   outImgctrl.SetImage( segmentation->ib );
 }
      }
    }
}
```

Segmentation.h :

```
#include "MainWindow.h"
#include <CtrlLib/CtrlLib.h>

class Segmentation {

public:
 typedef Segmentation CLASSNAME;

 Segmentation(  );
 // Does the evolution
 bool Iteration(  );
 // Called from MainWindow, generates the matrix of the image
```

```
 bool LoadImage( One < StreamRaster > streamRaster );

 ImageBuffer ib;

 // contains the intensities of the input image
 int **picture;

protected:
 void Init( One < StreamRaster > streamRaster );
};
```

Segmentation.cpp :

```
Segmentation::Segmentation(  )
{
 imageHeight = -1;
 imageWidth = -1;
 picture = NULL;
}

//---------------------------------

bool Segmentation::LoadImage( One < StreamRaster > r )
{
 imageHeight = r->GetHeight();
 imageWidth = r->GetWidth();

 Init( r );

 if ( (imageHeight != -1) && (imageWidth != -1) )
  return true;
 else
  return false;
}

//---------------------------------

void Segmentation::Init( One < StreamRaster > r )
{
 // memory allocations
 picture = new int* [imageWidth];
 for ( int i = 0; i < imageWidth; i++ )
  picture[i] = new int[imageHeight]; // the image intensities

 // image -> matrix
 for(int i = 0; i < imageHeight; i++) {
```

```
  RasterLine l = r->GetLine(i);
  for(int j = 0; j < imageWidth; j++) {
   picture[i][j] = l[j].b;
  }
 }

        ...

 // matrix -> image

 ib(imageHeight, imageWidth);

 for(int y = 0; y < imageWidth; y++) {
     RGBA *l = ib[y];
     for(int x = 0; x < imageHeight; x++) {
                l->a = 255;
                l->r = picture[y][x];
                l->g = picture[y][x];
                l->b = picture[y][x];
                l++;
     }
 }

        //Premultiply(ib);
}
```

I hope this is enough. Thanks!

Subject: Re: Draw stuff
Posted by mrjt on Tue, 05 Feb 2008 10:03:42 GMT
View Forum Message <> Reply to Message

ib(imageHeight, imageWidth); is the form for initializing a variable. Since it is already initialised you should use:
ib.Create(width, height);

Subject: Re: Draw stuff
Posted by Indio on Tue, 05 Feb 2008 22:03:03 GMT
View Forum Message <> Reply to Message

Thanx, that's works. Now I try it on an iterative way.

Subject: Re: Draw stuff
Posted by Indio on Fri, 08 Feb 2008 15:02:10 GMT
View Forum Message <> Reply to Message

If I try like this, I only get the final image of the iteration. The images created from the results of the midsteps aren't shown.
It's only a test example.

```
int it;
Splitter splitter;
ImageCtrl outImgctrl;

Constructor(  )
{
  it = 1;
  splitter.Add(outImgctrl);
}

void Start(  )
{
 int itMaxNumber = 40;

 while( it < itMaxNumber )
 {
  MyDraw(  );
  it++;
 }
}

void MyDraw(  )
{
 ImageBuffer imgb(128, 128);

 for( int i = 0; i < 128; i++ )
 {
  RGBA *l = imgb[i];
  for( int j = 0; j < 128; j++ )
  {
                    // draws a line moving downwards
   if ( it == i )
   {
    l[j] = Black(  );
   }
   else
    l[j] = White(  );
  }
 }

 outImgctrl.SetImage( imgb );
```

```
 Refresh(  ); // needed? what's for?
}
```

What's wrong with this?

Subject: Re: Draw stuff
Posted by mrjt on Fri, 08 Feb 2008 16:00:47 GMT
View Forum Message <> Reply to Message

```
void Start(  )
{
 int itMaxNumber = 40;

 while( it < itMaxNumber )
 {
  MyDraw(  );
              ProcessEvents();
  it++;
 }
}
```

Refresh sends a Paint event (kind of), and you need to call ProcessEvents to catch it and redraw the screen before the next iteration (you may want a delay in there as well). In this case you don't need to call Refresh() because it's called in ImageCtrl::SetImage.

This might be a better solution though:

```
void Start()
{
   it = 0;
   SetTimeCallback(-50 THISBACK(MyDraw));
}

void MyDraw()
{
 ...
 if (it >= itMaxCount)
  KillTimeCallback();
}
```

Subject: Re: Draw stuff
Posted by Indio on Fri, 08 Feb 2008 17:09:58 GMT
View Forum Message <> Reply to Message

I'm grateful. I guess I won't need any delay, the counting in the iteration steps are quite complex, thus slow...

I know these were lame questions, but you helped me a lot. Thx!