
Subject: Re: saving a paletted PNG
Posted by [mirek](#) on Sun, 29 Oct 2006 23:34:44 GMT
[View Forum Message](#) <> [Reply to Message](#)

lindquist wrote on Sun, 29 October 2006 18:10 I'm having some more problems.
I have this test code:

```
void WritePNG(const String& filename)
{
    ImageBuffer ib(256, 256);
    ib.SetKind(IMAGE_OPAQUE);

    RGBA pal[256];
    for (int i=0; i<256; ++i)
    {
        pal[i] = Color(i,i,i);
    }

    int idx = 0;
    for (int y=0; y<256; ++y)
    {
        RGBA* line = ib[y];
        for (int x=0; x<256; ++x)
        {
            *line = pal[idx%256];
            ++line;
            ++idx;
        }
    }

    PNGEncoder png(8);
    png.SetPalette(pal);
    png.SaveFile(filename, ib);
}
```

I would assume this code would generate a nice horizontal gradient, but it doesn't. Instead I get something that seems a bit like it's converted to 5bit colors. (every 8 pixels it jumps 16 indexes in the palette)

That is unfortunately the limitation of palette conversion code - but in fact, we are not alone there... (almost any palette code I have studied before implementing U++ version did the same).

The problem is - you have 256 colors in palette and you need to find nearest color for 256*256*256 colors for tens of thousands pixels. If you would perform search for each pixel, you would be exporting each Image for hours.

Therefore you need something faster albeit less accurate. Usually, what you do is to cut some bits out and lookup index in map (you have to cut those bits because for full RGB you would need 16MB map). Works well except for single color gradients which is exactly the case you are trying....

Mirek
