

How to Change Pixel Color of an Image in C#.NET

Asked 6 years, 4 months agoActive 3 years, 9 months agoViewed 47k times

I am working with Images in Java, I have designed more over 100+ images(png) format, They were all Trasparent and Black Color Drawing.

The problem is, Now I have been asked to change the color of the Drawing (Black -to).

I have searched many code snipped at google,that changes the Bitmap (pixels) of the Image, but i am not guessing what i have to do to match the exact pixel and replace specially when the images if in Transparent mode. Below is the code in .Net (C#)

```
Bitmap newBitmap = new Bitmap(scrBitmap.Width, scrBitmap.Height);
for (int i = 0; i < scrBitmap.Width; i++)
{
    for (int j = 0; j < scrBitmap.Height; j++)
    {
        originalColor = scrBitmap.GetPixel(i, j);
        if(originalColor = Color.Black)
            newBitmap.SetPixel(i, j, Color.Red);
    }
}
return newBitmap;
```

but it was not matching at all, i debugged it, throughout the file, there was no value of Red,Green,Blue

Blog

Coding Salaries in 2019: Updating the Stack Overflow Salary Calculator

IMHO: The Mythical Fullstack Engineer

Featured on Meta

Official FAQ on gender pronouns and Code of Conduct changes

Yet another "step down as moderator" post

I'm resigning as a Stack Overflow Community Elected Moderator

Join Stack Overflow to learn, share knowledge, and build your career.

Sign up with email

Sign up with Google

Sign up with Facebook

×

c# .net drawing pixels bitmapimage

share improve this question

edited Jun 20 '13 at 9:34

asked Jun 20 '13 at 7:58

 Bibi Tahira
654 ● 4 ● 12 ● 38

add a comment

4 Answers

activeoldestvotes

Here is the Solution I have done with Pixels.

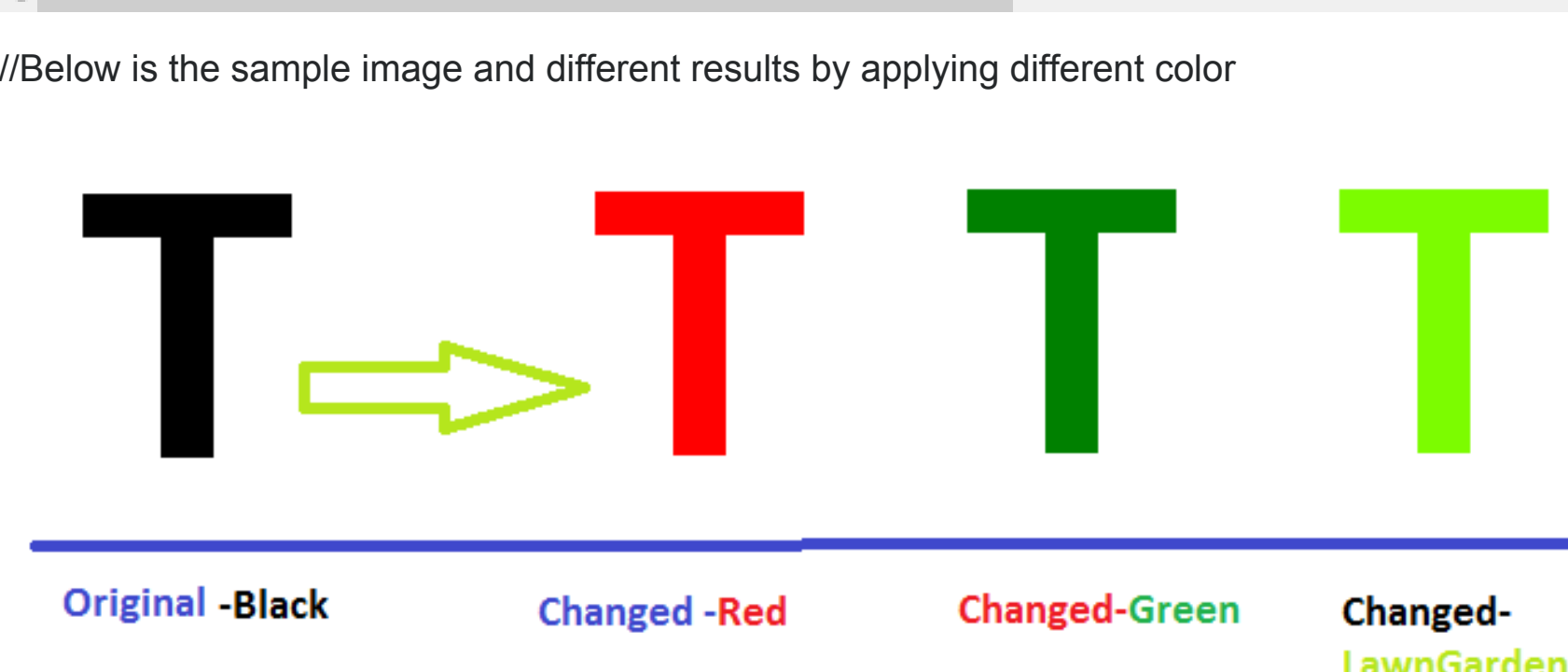
Attaching the source code so one can try the exact and get the result.

I have sample images of 128x128 (Width x Height).

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Drawing;
using System.IO;
//using System.Globalization;

namespace colorchange
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                Bitmap bmp = null;
                //The Source Directory in debug\bin\Big\
                string[] files = Directory.GetFiles("Big\\");
                foreach (string filename in files)
                {
                    bmp = (Bitmap)Image.FromFile(filename);
                    bmp = ChangeColor(bmp);
                    string[] splitter = filename.Split('\\');
                    //Destination Directory debug\bin\BigGreen\
                    bmp.Save("BigGreen\\" + splitter[1]);
                }
            }
            catch (System.Exception ex)
            {
                Console.WriteLine(ex.ToString());
            }
        }
        public static Bitmap ChangeColor(Bitmap scrBitmap)
        {
            //You can change your new color here. Red,Green,LawnGreen any..
        }
    }
}
```

//Below is the sample image and different results by applying different color





Code modifications will be highly appreciated.

share improve this answer

edited Dec 30 '15 at 8:34

answered Jun 20 '13 at 8:53

 cup
4,892 ● 2 ● 10 ● 29

 DareDevil
4,081 ● 5 ● 37 ● 74

1 This is the best reply and I am sure my requirement is fulfilled, thanks to DareDevil – Bibi Tahira Jun 20 '13 at 8:57

4 Well this will replace every color to the new one (not just a selected one) and check on alpha will produce sub-optimal results if there is a gradient but if it satisfies OP... – Adriano Repetti Jun 20 '13 at 9:34

1 well this code is fine for images of single color. – DareDevil Jun 20 '13 at 13:02

Changed actula to acutal in the code – cup Dec 30 '15 at 8:34

@DareDevil thanks. You code to change colour only object in the image. But I only want to change the color of the background image, not change a colour of the object. My origin image: imgur.com/LphfKdV You changed: imgur.com/DJc0NOU I want like: imgur.com/4pvWip8 How to do this? Thanks. – vanloc May 26 '16 at 8:08

add a comment

Before we talk about performance let's check your code:

```
var originalColor = scrBitmap.GetPixel(i, j);
if (originalColor = Color.Black)
    newBitmap.SetPixel(i, j, Color.Red);
```

Here there are two errors:

1. You do not compare to `Color.Black` but you **assign** `Color.Black` to `originalColor`.
2. You do not handle transparency.

To check for transparency you should compare not the `Color` object but the R, G, B values, let's change to:

```
var originalColor = scrBitmap.GetPixel(i, j);
if (originalColor.R == 0 && originalColor.G == 0 && originalColor.B == 0)
    newBitmap.SetPixel(i, j, Color.FromArgb(originalColor.A, Color.Red));
```

Now you'll see that it works but it takes a very long time to process each image: `GetPixel` and `SetPixel` are pretty slow (primary because they check and calculate everything for each call). It's much better to handle bitmap data directly. If you know the image format in advance (and it's fixed for each image) then you can do it much much faster with little bit more code:

```
static unsafe Bitmap ReplaceColor(Bitmap source,
    Color toReplace,
    Color replacement)
{
    const int pixelSize = 4; // 32 bits per pixel

    Bitmap target = new Bitmap(
        source.Width,
        source.Height,
        PixelFormat.Format32bppArgb);

    BitmapData sourceData = null, targetData = null;

    try
    {
        sourceData = source.LockBits(
            new Rectangle(0, 0, source.Width, source.Height),
            ImageLockMode.ReadOnly, PixelFormat.Format32bppArgb);

        targetData = target.LockBits(
            new Rectangle(0, 0, target.Width, target.Height),
            ImageLockMode.WriteOnly, PixelFormat.Format32bppArgb);

        for (int y = 0; y < source.Height; ++y)
        {
            byte* sourceRow = (byte*)sourceData.Scan0 + (y * sourceData.Stride);
            byte* targetRow = (byte*)targetData.Scan0 + (y * targetData.Stride);

            for (int x = 0; x < source.Width; ++x)
            {
                byte b = sourceRow[x * pixelSize + 0];
                byte g = sourceRow[x * pixelSize + 1];
                byte r = sourceRow[x * pixelSize + 2];
                byte a = sourceRow[x * pixelSize + 3];

                if (toReplace.R == r && toReplace.G == g && toReplace.B == b)
                {

```

Of course this can be [further optimized](#) and you may need to handle different formats ([see this list of pixel formats](#) and [this article](#) about their layout) but consider it a starting point to work with bitmaps.

For completeness this is equivalent color without direct access to bitmap data. Please note that this should be rarely used because it's terribly slow.

```
static Bitmap ReplaceColor(Bitmap source,
    Color toReplace,
    Color replacement)
{
    var target = new Bitmap(source.Width, source.Height);

    for (int x = 0; x < source.Width; ++x)
    {
        for (int y = 0; y < source.Height; ++y)
        {
            var color = source.GetPixel(x, y);
            target.SetPixel(x, y, color == toReplace ? replacement : color);
        }
    }

    return target;
}
```

Also please note that this consider alpha channel in comparison (so 50% transparent green, for example, is not same color as 30% transparent green). To ignore alpha you may use something like this:


```
if (color.R == toReplace.R && color.G == toReplace.G && color.B == toReplace.B)
```

Finally if you know that pixels to replace are little you may create a raw copy of original image (using `Graphics.FromImage` to create a context and to draw into it `source` bitmap), in such way you'll call `SetPixel()` only when there is a replacement. IMO any optimization here is pretty useless: if you need performance use first solution...

share improve this answer

edited Aug 13 '14 at 11:59

answered Jun 20 '13 at 8:02

 Adriano Repetti
52,7k ● 15 ● 111 ● 172

In Hurry, I did all and it got deleted == – Bibi Tahira Jun 20 '13 at 8:58

1 @BibiTahira what got deleted? – Adriano Repetti Jun 20 '13 at 9:02

when i was coparing my pixel color with system color – Bibi Tahira Jun 20 '13 at 9:26

@BibiTahira see code examples, you *mustn't* compare Color objects but individual components (because of alpha in the original image). – Adriano Repetti Jun 20 '13 at 9:35

1 Thank you sir. I got your point. – Bibi Tahira Jun 20 '13 at 10:05

show 6 more comments

One way to efficiently replace a color is to use a remap table. In the following example, an image is drawn inside a picture box. In the Paint event, the color Color.Black is changed to Color.Blue:

```
private void pictureBox_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;
    using (Bitmap bmp = new Bitmap("myImage.png"))
    {
        // Set the image attribute's color mappings
        ColorMap[] colorMap = new ColorMap[1];
        colorMap[0] = new ColorMap();
        colorMap[0].OldColor = Color.Black;
        colorMap[0].NewColor = Color.Blue;
        ImageAttributes attr = new ImageAttributes();
        attr.SetRemapTable(colorMap);
        // Draw using the color map
        Rectangle rect = new Rectangle(0, 0, bmp.Width, bmp.Height);
        g.DrawImage(bmp, rect, 0, 0, rect.Width, rect.Height, GraphicsUnit.Pixel, attr);
    }
}
```

More information: <http://msdn.microsoft.com/en-us/library/4b4dc1kz%28v=vs.110%29.aspx>

share improve this answer

answered Nov 24 '14 at 9:29

 Nicholas
1,589 ● 1 ● 16 ● 23

I will give you another solution since this does not calculate for every pixel.

Its short and simple. Convert time is 62 ms:

```
public Bitmap Color(Bitmap original)
{
    //create a blank bitmap the same size as original
    Bitmap newBitmap = new Bitmap(original.Width, original.Height);

    //get a graphics object from the new Image
    Graphics g = Graphics.FromImage(newBitmap);

    //create the color you want ColorMatrix
    //now is set to red, but with different values
    //you can get anything you want.
    ColorMatrix colorMatrix = new ColorMatrix(
        new float[][]
        {
            new float[] {1f, .0f, .0f, 0, 0},
            new float[] {1f, .0f, .0f, 0, 0},
            new float[] {1f, .0f, .0f, 0, 0},
            new float[] {0, 0, 1, 0, 0},
            new float[] {0, 0, 0, 0, 1}
        });

    //create some image attributes
    ImageAttributes attributes = new ImageAttributes();

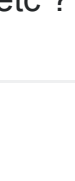
    //set the color matrix attribute
    attributes.SetColorMatrix(colorMatrix);

    //draw original image on the new image using the color matrix
    g.DrawImage(original, new Rectangle(0, 0, original.Width, original.Height),
        0, 0, original.Width, original.Height, GraphicsUnit.Pixel, attributes);

    //release sources used
    g.Dispose();
    return newBitmap;
}
```

share improve this answer

answered Mar 2 '15 at 22:43





 Nicholas
1,589 ● 1 ● 16 ● 23






2 How can you create a color matrix for a particular color ? Such as Green, Blue etc ? – Ajay Sharma Oct 26 '15 at 12:51

add a comment

Your Answer

B /





Sign up or log in

Sign up using Google

Sign up using Facebook

Sign up using Email and Password

Post as a guest

Name

Email

Required, but never shown

Post Your Answer

By clicking "Post Your Answer", you agree to our [terms of service](#), [privacy policy](#) and [cookie policy](#)

Not the answer you're looking for? Browse other questions tagged c# .net drawing pixels

bitmapimage or ask your own question.



STACK OVERFLOW

Questions
Jobs
Developer Jobs Directory
Salary Calculator
Help
Mobile
Disable Responsiveness






PRODUCTS

Teams
Talent
Advertising
Enterprise

COMPANY

About
Press
Work Here
Legal
Privacy Policy
Contact Us

STACK EXCHANGE NETWORK

Technology 
Life / Arts 
Culture / Recreation 
Science 
Other 

Blog Facebook Twitter LinkedIn

site design / logo © 2019 Stack Exchange Inc; user contributions licensed under cc by-sa 4.0 with attribution required.
rev 2019.10.18.35513