Algorithm to randomly generate an aesthetically-pleasing color palette [closed]

Asked 15 years, 1 month ago Modified 7 months ago Viewed 152k times

329

Closed. This question needs to be more focused. It is not currently accepting answers.

Want to improve this question? Update the question so it focuses on one problem only by editing this post.

Closed 6 years ago.

Improve this question

I'm looking for a simple algorithm to generate a large number of random, aesthetically pleasing colors. So no crazy neon colors, colors reminiscent of feces, etc.

I've found solutions to this problem but they rely on alternative color palettes than RGB. I would rather just use straight RGB than mapping back and forth. These other solutions also can at most generate only 32 or so pleasing random colors.

Any ideas would be great.

algorithm colors

Share Edit Follow

edited Feb 11, 2013 at 3:27

Alex L

8,758 6 49 75

asked Sep 4, 2008 at 1:54

Brian Gianforcaro

26.6k 11 58 77

You could check out HSL and HSV. – Jim Keener Feb 15, 2011 at 20:35

1 That's not a algorithm. – Matt The Ninja Aug 23, 2016 at 8:06

I'll add a comment since question is closed. I haven't tried it, but it would be interesting to try a fractal or procedural way, using i.e. Midpoint Displacement algorithm or variants (lighthouse3d.com/opengl/terrain/index.php?mpd2) or Perlin Noise, but instead of heights to mix the rgb components of the colors. – alfoks Apr 24, 2017 at 8:12

Take a look at the color palette generator I have built and it's source code: hypejunction.github.io/color-wizard You can basically take a range of hues, determine luminosity steps that result in colors you prefer and determine saturation using parabolic formula. – hypeJunction Apr 8, 2019 at 9:58

related - stackoverflow.com/q/1664140/104380 – vsync Apr 25, 2019 at 20:30

Highest score (default)

16 Answers

Sorted by:

455

You could average the RGB values of random colors with those of a constant color:

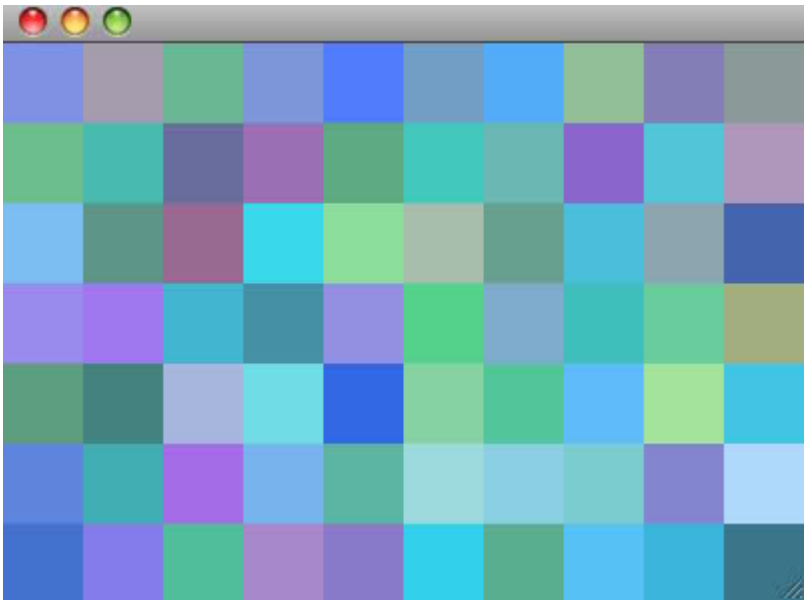
(example in Java)

```
public Color generateRandomColor(Color mix) {
  Random random = new Random();
  int red = random.nextInt(256);
  int green = random.nextInt(256);
  int blue = random.nextInt(256);

  // mix the color
  if (mix != null) {
      red = (red + mix.getRed()) / 2;
      green = (green + mix.getGreen()) / 2;
      blue = (blue + mix.getBlue()) / 2;
  }

  Color color = new Color(red, green, blue);
  return color;
}
```

Mixing random colors with white (255, 255, 255) creates neutral pastels by increasing the lightness while keeping the hue of the original color. These randomly generated pastels usually go well together, especially in large numbers.

Here are some pastel colors generated using the above method:

You could also mix the random color with a constant pastel, which results in a tinted set of neutral colors. For example, using a light blue creates colors like these:



Going further, you could add heuristics to your generator that take into account complementary colors or levels of shading, but it all depends on the impression you want to achieve with your random colors.

Some additional resources:

- http://en.wikipedia.org/wiki/Color_theory
- http://en.wikipedia.org/wiki/Complementary_color

Share Edit Follow

edited Feb 14 at 7:03

Gangula
5,261 4 31 60

answered Sep 4, 2008 at 6:00

David Crow
16.1k 8 42 34

5 The only caveat seems to be the loss of distinguishability. By taking average of red green and blue with one color such as white, naturally all colors move closer to white and reduce the differences between them. For example if you look at the first screenshot: When those greens are next to each other, surely I can distinguish them, but what if they're not adjacent in some produced graphic? Could be problematic. – Zelphir Kaltstahl Mar 19, 2016 at 2:03

When generating random color component, try restricting upper bound with something less, like 200. – Nickolodeon Oct 9, 2018 at 17:14

91

I would use a color wheel and given a random position you could add the golden angle (137,5 degrees)

http://en.wikipedia.org/wiki/Golden_angle

in order to get different colours each time that do not overlap.

Adjusting the brightness for the color wheel you could get also different bright/dark color combinations.

I've found this blog post that explains really well the problem and the solution using the golden ratio.

http://martin.ankerl.com/2009/12/09/how-to-create-random-colors-programmatically/

UPDATE: I've just found this other approach:

It's called RYB(red, yellow, blue) method and it's described in this paper:

http://threekings.tk/mirror/ryb_TR.pdf

as "Paint Inspired Color Compositing".

The algorithm generates the colors and each new color is chosen to maximize its euclidian distance to the previously selected ones.

Here you can find a a good implementation in javascript:

http://afriggeri.github.com/RYB/

UPDATE 2:

The Sciences Po Medialb have just released a tool called "I want Hue" that generate color palettes for data scientists. Using different color spaces and generating the palettes by using k-means clustering or force vectors (repulsion graphs) The results from those methods are very good, they show the theory and an implementation in their web page.

http://tools.medialab.sciences-po.fr/iwanthue/index.php

Share Edit Follow

edited Feb 12, 2013 at 11:26

answered Feb 15, 2011 at 19:01

Fgblanch

5,195 8 37 51

24

In javascript:

```
function pastelColors(){
  var r = (Math.round(Math.random()* 127) + 127).toString(16);
  var g = (Math.round(Math.random()* 127) + 127).toString(16);
  var b = (Math.round(Math.random()* 127) + 127).toString(16);
  return '#' + r + g + b;
}
```

Saw the idea here: http://blog.functionalfun.net/2008/07/random-pastel-colour-generator.html

Share Edit Follow

edited Feb 14 at 7:04

Gangula

5,261 4 31 60

answered Sep 4, 2012 at 15:05

motobói

1,687 18 24

Not completely correct: you need to zero-padding for single-digit numbers too... Anyways, here is a working fiddle for lookers: jsfiddle.net/RedDevil/LLYBQ ----- Scratch that... I didn't notice the +127 bit... But then this won't generate dark shades. – kumarharsh Jul 4, 2014 at 11:12

3 The original idea was to generate arbitrarily pleasant colors. The dark shades don't seem pleasant to me ;) – motobói Jul 4, 2014 at 14:04

check this link for JavaScript - stackoverflow.com/a/54279008/6908282 – Gangula Feb 14 at 8:52

11

Converting to another palette is a far superior way to do this. There's a reason they do that: other palettes are 'perceptual' - that is, they put similar seeming colors close together, and adjusting one variable changes the color in a predictable manner. None of that is true for RGB, where there's no obvious relationship between colors that "go well together".

Share Edit Follow

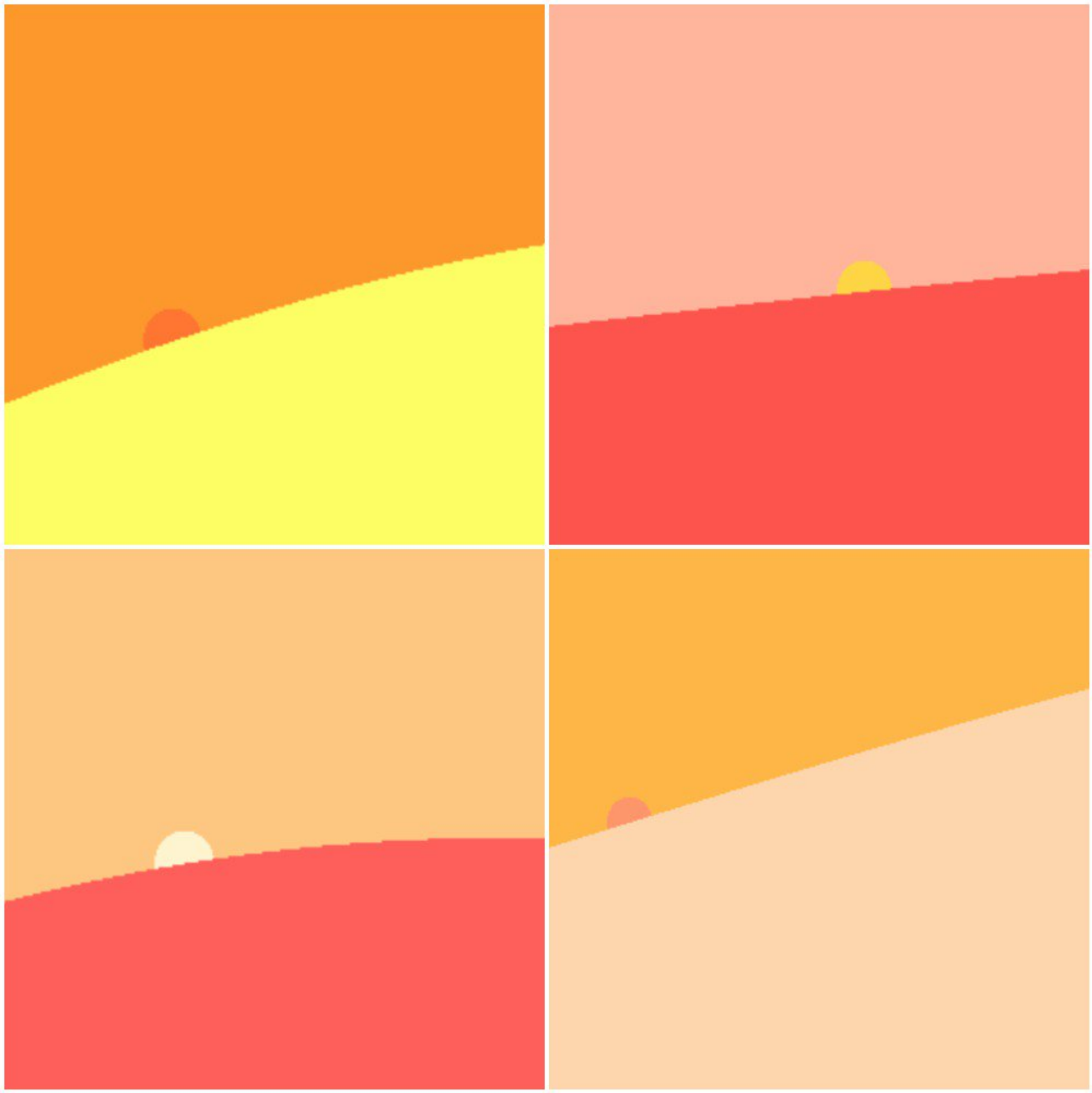
answered Jun 7, 2009 at 20:45

Nick Johnson
101k 16 129 198

9

I've had success using TriadMixing and CIE94 to avoid similar colors. The following image uses input colors red, yellow, and white. See here.

```
// http://devmag.org.za/2012/07/29/how-to-choose-colours-procedurally-
algorithms/#:~:text=120%20and%20240.-,7.%20Triad%20Mixing,-
This%20algorithm%20takes

public static Color RandomMix(Color color1, Color color2, Color color3,
   float greyControl)
{
   int randomIndex = random.NextByte() % 3;
   float mixRatio1 =
      (randomIndex == 0) ? random.NextFloat() * greyControl :
random.NextFloat();
   float mixRatio2 =
      (randomIndex == 1) ? random.NextFloat() * greyControl :
random.NextFloat();
   float mixRatio3 =
      (randomIndex == 2) ? random.NextFloat() * greyControl :
random.NextFloat();
   float sum = mixRatio1 + mixRatio2 + mixRatio3;

   mixRatio1 /= sum;
   mixRatio2 /= sum;
   mixRatio3 /= sum;

   return Color.FromArgb(
      255,
      (byte)(mixRatio1 * color1.R + mixRatio2 * color2.R + mixRatio3 *
color3.R),
      (byte)(mixRatio1 * color1.G + mixRatio2 * color2.G + mixRatio3 *
color3.G),
      (byte)(mixRatio1 * color1.B + mixRatio2 * color2.B + mixRatio3 *
color3.B));
}
```



Share Edit Follow

edited Feb 14 at 7:09

Gangula
5,261 4 31 60

answered Dec 9, 2016 at 20:15

Moriarty
3,967 1 31 27

5

An answer that shouldn't be overlooked, because it's simple and presents advantages, is sampling of real life photos and paintings. sample as many random pixels as you want random colors on thumbnails of modern art pics, cezanne, van gogh, monnet, photos... the advantage is that you can get colors by theme and that they are organic colors. just put 20 - 30 pics in a folder and random sample a random pic every time.

Conversion to HSV values is a widespread code algorithm for psychologically based palette. hsv is easier to randomize.

Share Edit Follow

answered Aug 16, 2013 at 8:59

bandybabboon
2,218 1 23 33

1 mkweb.bcgsc.ca/color_summarizer/?analyze Here is an online tool that can analyse real-life photos and returns to you Grapher of the RGB values say can get a feel of what real-life photos have in their graphs.... extremely useful website, which is essential if you are trying to design avant-garde algorithms for random colours- – bandybabboon Sep 26, 2013 at 19:10

4

Use distinct-colors.

Written in javascript.

It generates a palette of *visually* distinct colors.

distinct-colors is highly configurable:

- Choose how many colors are in the palette
- Restrict the hue to a specific range
- Restrict the chroma (saturation) to a specific range
- Restrict the lightness to a specific range
- Configure general quality of the palette

Share Edit Follow

answered Aug 20, 2015 at 17:24
InternalFX
1,475 12 14

4

In php:

```
function pastelColors() {
    $r = dechex(round(((float) rand() / (float) getrandmax()) * 127) + 127);
    $g = dechex(round(((float) rand() / (float) getrandmax()) * 127) + 127);
    $b = dechex(round(((float) rand() / (float) getrandmax()) * 127) + 127);

    return "#" . $r . $g . $b;
}
```

source: https://stackoverflow.com/a/12266311/2875783

Share Edit Follow

edited May 23, 2017 at 12:18
Community Bot
1 1

answered Mar 12, 2014 at 20:25
Petr Bugyík
498 4 12

3

Here is quick and dirty color generator in C# (using 'RYB approach' described in this article). It's a rewrite from JavaScript.

Use:

```
List<Color> ColorPalette = ColorGenerator.Generate(30).ToList();
```

First two colors tend to be white and a shade of black. I often skip them like this (using Linq):

```
List<Color> ColorsPalette = ColorGenerator
            .Generate(30)
            .Skip(2) // skip white and black
            .ToList();
```

Implementation:

```
public static class ColorGenerator
{

    // RYB color space
    private static class RYB
    {
        private static readonly double[] White = { 1, 1, 1 };
        private static readonly double[] Red = { 1, 0, 0 };
        private static readonly double[] Yellow = { 1, 1, 0 };
        private static readonly double[] Blue = { 0.163, 0.373, 0.6 };
        private static readonly double[] Violet = { 0.5, 0, 0.5 };
        private static readonly double[] Green = { 0, 0.66, 0.2 };
        private static readonly double[] Orange = { 1, 0.5, 0 };
        private static readonly double[] Black = { 0.2, 0.094, 0.0 };

        public static double[] ToRgb(double r, double y, double b)
        {
            var rgb = new double[3];
            for (int i = 0; i < 3; i++)
            {
                rgb[i] = White[i]  * (1.0 - r) * (1.0 - b) * (1.0 - y) +
                         Red[i]    * r         * (1.0 - b) * (1.0 - y) +
                         Blue[i]   * (1.0 - r) * b         * (1.0 - y) +
                         Violet[i] * r         * b         * (1.0 - y) +
                         Yellow[i] * (1.0 - r) * (1.0 - b) *        y +
                         Orange[i] * r         * (1.0 - b) *        y +
                         Green[i]  * (1.0 - r) * b         *        y +
                         Black[i]  * r         * b         *        y;
            }

            return rgb;
        }
    }

    private class Points : IEnumerable<double[]>
    {
        private readonly int pointsCount;
        private double[] picked;
```

Share Edit Follow

edited Jan 26, 2014 at 9:56

answered Jan 26, 2014 at 9:09

m1ch4ls

3,337 18 31

I have removed the Java answer but if required, the Java version can be seen in this Gist: gist.github.com/lotsabackscatter/3f6a658fd7209e010dad – Dylan Watson Jun 11, 2015 at 16:10

3

David Crow's method in an R two-liner:

```
GetRandomColours <- function(num.of.colours, color.to.mix=c(1,1,1)) {
  return(rgb((matrix(runif(num.of.colours*3),
nrow=num.of.colours)*color.to.mix)/2))
}
```

Share Edit Follow

answered Feb 13, 2015 at 13:40

Dave_L

363 2 11

2

```
function fnGetRandomColour(iDarkLuma, iLightLuma)
{
  for (var i=0;i<20;i++)
  {
    var sColour = ('ffffff' + Math.floor(Math.random() *
0xFFFFFF).toString(16)).substr(-6);

    var rgb = parseInt(sColour, 16);   // convert rrggbb to decimal
    var r = (rgb >> 16) & 0xff;  // extract red
    var g = (rgb >>  8) & 0xff;  // extract green
    var b = (rgb >>  0) & 0xff;  // extract blue

    var iLuma = 0.2126 * r + 0.7152 * g + 0.0722 * b; // per ITU-R BT.709


    if (iLuma > iDarkLuma && iLuma < iLightLuma) return sColour;
  }
  return sColour;
}
```

For pastel, pass in higher luma dark/light integers - ie fnGetRandomColour(120, 250)

Credits: all credits to http://paulirish.com/2009/random-hex-color-code-snippets/
stackoverflow.com/questions/12043187/how-to-check-if-hex-color-is-too-black

Share Edit Follow

answered Feb 11, 2013 at 10:28

ChilledFlame

107 1 2

1

JavaScript adaptation of David Crow's original answer, IE and Nodejs specific code included.

```
generateRandomComplementaryColor = function(r, g, b){
    //--- JavaScript code
    var red = Math.floor((Math.random() * 256));
    var green = Math.floor((Math.random() * 256));
    var blue = Math.floor((Math.random() * 256));
    //---

    //--- Extra check for Internet Explorers, its Math.random is not random
enough.
```

```
    if(!/MSIE 9/i.test(navigator.userAgent) && !/MSIE
10/i.test(navigator.userAgent) && !/rv:11.0/i.test(navigator.userAgent)){
        red = Math.floor((('0.' + window.crypto.getRandomValues(new
Uint32Array(1))[0]) * 256));
        green = Math.floor((('0.' + window.crypto.getRandomValues(new
Uint32Array(1))[0]) * 256));
        blue = Math.floor((('0.' + window.crypto.getRandomValues(new
Uint32Array(1))[0]) * 256));
    };
    //---

    //--- nodejs code
    /*
    crypto = Npm.require('crypto');
    red = Math.floor((parseInt(crypto.randomBytes(8).toString('hex'), 16)) *
1.0e-19 * 256);
    green = Math.floor((parseInt(crypto.randomBytes(8).toString('hex'), 16)) *
1.0e-19 * 256);
    blue = Math.floor((parseInt(crypto.randomBytes(8).toString('hex'), 16)) *
1.0e-19 * 256);
    */
    //---

    red = (red + r)/2;
    green = (green + g)/2;
    blue = (blue + b)/2;

    return 'rgb(' + Math.floor(red) + ', ' + Math.floor(green) + ', ' +
```

Run the function using:

```
generateRandomComplementaryColor(240, 240, 240);
```

Share Edit Follow

answered Jul 25, 2015 at 17:44

Sceptic
1,659 2 15 25

0

I'd strongly recommend using a CG HSVtoRGB shader function, they are awesome... it gives you natural color control like a painter instead of control like a crt monitor, which you arent presumably!

This is a way to make 1 float value. i.e. Grey, into 1000 ds of combinations of color and brightness and saturation etc:

```
int rand = a global color randomizer that you can control by script/ by a
crossfader etc.
float h = perlin(grey,23.3*rand)
float s = perlin(grey,54,4*rand)
float v = perlin(grey,12.6*rand)

Return float4 HSVtoRGB(h,s,v);
```

result is AWESOME COLOR RANDOMIZATION! it's not natural but it uses natural color gradients and it looks organic and controlleably irridescent / pastel parameters.

For perlin, you can use this function, it is a fast zig zag version of perlin.

```
function  zig ( xx : float ): float{    //lfo nz -1,1
    xx= xx+32;
    var x0 = Mathf.Floor(xx);
    var x1 = x0+1;
    var v0 = (Mathf.Sin (x0*.014686)*31718.927)%1;
    var v1 = (Mathf.Sin  (x1*.014686)*31718.927)%1;
    return Mathf.Lerp( v0 , v1 , (xx)%1 )*2-1;
}
```

Share Edit Follow

edited Apr 1, 2014 at 21:43

answered Mar 30, 2014 at 12:23

bandybabboon

2,218 1 23 33

0

It's going to be hard to get what you want algorithmically - people have been studying color theory for a long time, and they don't even know all the rules.

However, there are *some* rules which you can use to cull bad color combinations (ie, there are rules for clashing colors, and choosing complementary colors).

I'd recommend you visit your library's art section and check out books on color theory to gain a better understanding of what is a good color before you try to make one - it appears you might not even know why certain combinations work and others don't.

-Adam

Share Edit Follow

answered Sep 4, 2008 at 3:36

Adam Davis

92.1k 60 264 330

0

you could have them be within a certain brightness. that would control the ammount of "neon" colors a bit. for instance, if the "brightness"

```
brightness = sqrt(R^2+G^2+B^2)
```

was within a certain high bound, it would have a washed out, light color to it. Conversely, if it was within a certain low bound, it would be darker. This would eliminate any crazy, standout colors and if you chose a bound really high or really low, they would all be fairly close to either white or black.

Share Edit Follow

answered Sep 4, 2008 at 2:16

helloandre

10.6k 8 47 64

0

Here is something I wrote for a site I made. It will auto-generate a random flat background-color for any div with the class `.flat-color-gen` . Jquery is only required for the purposes

of adding css to the page; it's not required for the main part of this, which is the `generateFlatColorWithOrder()` method.

JsFiddle Link

```
(function($) {
    function generateFlatColorWithOrder(num, rr, rg, rb) {
        var colorBase = 256;
        var red = 0;
        var green = 0;
        var blue = 0;
        num = Math.round(num);
        num = num + 1;
        if (num != null) {

            red = (num*rr) % 256;
            green = (num*rg) % 256;
            blue = (num*rb) % 256;
        }
        var redString = Math.round((red + colorBase) / 2).toString();
        var greenString = Math.round((green + colorBase) / 2).toString();
        var blueString = Math.round((blue + colorBase) / 2).toString();
        return "rgb("+redString+", "+greenString+", "+blueString+")";
        //return '#' + redString + greenString + blueString;
    }

    function generateRandomFlatColor() {
        return generateFlatColorWithOrder(Math.round(Math.random()*127));
    }

    var rr = Math.round(Math.random()*1000);
    var rg = Math.round(Math.random()*1000);
    var rb = Math.round(Math.random()*1000);
    console.log("random red: "+ rr);
    console.log("random green: "+ rg);
    console.log("random blue: "+ rb);
    console.log("-------------------------------------------------");
    $('.flat-color-gen').each(function(i, obj) {
        console.log(generateFlatColorWithOrder(i));
        $(this).css("background-color",generateFlatColorWithOrder(i, rr, rg,
rb).toString());
    });
})(window.jQuery);
```

Share Edit Follow

edited Jul 27, 2015 at 14:56

answered Jul 23, 2015 at 17:30

Ephraim

8,362 9 31 49