

Reference

Creating & Reading

setAlpha()

setBlue()

setGreen()

setRed()

toString()

alpha()

blue()

brightness()

color()

green()

hue()

lerpColor()

lightness()

p5.Color

paletteLerp()

red()

saturation()

Shape

Reference > color()

color()

Creates a **p5.Color** object.

By default, the parameters are interpreted as RGB values. Calling `color(255, 204, 0)` will return a bright yellow color. The way these parameters are interpreted may be changed with the `colorMode()` function.


The version of `color()` with one parameter interprets the value one of two ways. If the parameter is a number, it's interpreted as a grayscale value. If the parameter is a string, it's interpreted as a CSS color string.

The version of `color()` with two parameters interprets the first one as a grayscale value. The second parameter sets the alpha (transparency) value.

The version of `color()` with three parameters interprets them as RGB, HSB, or HSL colors, depending on the current `colorMode()`.

The version of `color()` with four parameters interprets them as RGBA, HSBA, or HSLA colors, depending on the current `colorMode()`. The last parameter sets the alpha (transparency) value.

Examples



▶

■


```
function setup() {
  createCanvas(100, 100);

  background(200);

  // Create a p5.Color object using RGB values.
  let c = color(255, 204, 0);

  // Draw the square.
  fill(c);
  noStroke();
  square(30, 20, 55);

  describe('A yellow square on a gray canvas.');
```



▶

■

```
function setup() {
  createCanvas(100, 100);

  background(200);


  // Create a p5.Color object using RGB values.
  let c1 = color(255, 204, 0);

  // Draw the left circle.
  fill(c1);
  noStroke();
  circle(25, 25, 80);

  // Create a p5.Color object using a grayscale value.
  let c2 = color(65);

  // Draw the right circle.
  fill(c2);
  circle(75, 75, 80);

  describe(
    'Two circles on a gray canvas. The circle in the top-left corner is yellow and the one at the bottom-right is gray.'
  );
}
```



▶

■


```
function setup() {
  createCanvas(100, 100);

  background(200);

  // Create a p5.Color object using a named color.
  let c = color('magenta');

  // Draw the square.
  fill(c);
  noStroke();
  square(20, 20, 60);

  describe('A magenta square on a gray canvas.');
```



▶

■

```
function setup() {
  createCanvas(100, 100);

  background(200);


  // Create a p5.Color object using a hex color code.
  let c1 = color('#0f0');

  // Draw the left rectangle.
  fill(c1);
  noStroke();
  rect(0, 10, 45, 80);

  // Create a p5.Color object using a hex color code.
  let c2 = color('#00ff00');

  // Draw the right rectangle.
  fill(c2);
  rect(55, 10, 45, 80);

  describe('Two bright green rectangles on a gray canvas.');
```



▶

■

```
function setup() {
  createCanvas(100, 100);

  background(200);

  // Create a p5.Color object using a RGB color string.
  let c1 = color('rgb(0, 0, 255)');

  // Draw the top-left square.
  fill(c1);
  square(10, 10, 35);


  // Create a p5.Color object using a RGB color string.
  let c2 = color('rgb(0%, 0%, 100%)');

  // Draw the top-right square.
  fill(c2);
  square(55, 10, 35);

  // Create a p5.Color object using a RGBA color string.
  let c3 = color('rgba(0, 0, 255, 1)');

  // Draw the bottom-left square.
  fill(c3);
  square(10, 55, 35);

  // Create a p5.Color object using a RGBA color string.
  let c4 = color('rgba(0%, 0%, 100%, 1)');
```



▶

■

```
function setup() {
  createCanvas(100, 100);

  background(200);


  // Create a p5.Color object using a HSL color string.
  let c1 = color('hsl(160, 100%, 50%)');

  // Draw the left rectangle.
  noStroke();
  fill(c1);
  rect(0, 10, 45, 80);

  // Create a p5.Color object using a HSLA color string.
  let c2 = color('hsla(160, 100%, 50%, 0.5)');

  // Draw the right rectangle.
  fill(c2);
  rect(55, 10, 45, 80);

  describe('Two sea green rectangles. A darker rectangle on the left and a brighter one on the right.');
```



▶

■

```
function setup() {
  createCanvas(100, 100);

  background(200);


  // Create a p5.Color object using a HSB color string.
  let c1 = color('hsb(160, 100%, 50%)');

  // Draw the left rectangle.
  noStroke();
  fill(c1);
  rect(0, 10, 45, 80);

  // Create a p5.Color object using a HSBA color string.
  let c2 = color('hsba(160, 100%, 50%, 0.5)');

  // Draw the right rectangle.
  fill(c2);
  rect(55, 10, 45, 80);

  describe('Two green rectangles. A darker rectangle on the left and a brighter one on the right.');
```



▶

■

```
function setup() {
  createCanvas(100, 100);

  background(200);

  // Create a p5.Color object using RGB values.
  let c1 = color(50, 55, 100);

  // Draw the left rectangle.
  fill(c1);
  rect(0, 10, 45, 80);

  // Switch the color mode to HSB.
  colorMode(HSB, 100);

  // Create a p5.Color object using HSB values.
  let c2 = color(50, 55, 100);

  // Draw the right rectangle.
  fill(c2);
  rect(55, 10, 45, 80);

  describe('Two blue rectangles. A darker rectangle on the left and a brighter one on the right.');
```

Syntax

color(gray, [alpha])

color(v1, v2, v3, [alpha])

color(value)

color(values)

color(color)

Parameters

gray	Number: number specifying value between white and black.
alpha	Number: alpha value relative to current color range (default is 0-255).
v1	Number: red or hue value relative to the current color range.
v2	Number: green or saturation value relative to the current color range.
v3	Number: blue or brightness value relative to the current color range.
value	String: a color string.
values	Number[]: an array containing the red, green, blue, and alpha components of the color.
color	p5.Color:

Returns

p5.Color: resulting color.

This page is generated from the comments in [src/color/creating_reading.js](#). Please feel free to edit it and submit a pull request!

Related References

setAlpha Sets the alpha (transparency) value of a color.	setBlue Sets the blue component of a color.	setGreen Sets the green component of a color.	setRed Sets the red component of a color.
--------------------------------------------------------------------	-------------------------------------------------------	---------------------------------------------------------	-----------------------------------------------------