

RoutinesRGB

v1.9.2

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Module Index</b>	<b>1</b>
1.1	Modules . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Module Documentation</b>	<b>7</b>
4.1	Getters and Setters . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.1.2	Function Documentation . . . . .	7
4.1.2.1	blinkSpeed(uint8_t blinkSpeed) . . . . .	7
4.1.2.2	blue(uint16_t i) . . . . .	8
4.1.2.3	brightness(uint8_t brightness) . . . . .	8
4.1.2.4	color(uint16_t i) . . . . .	8
4.1.2.5	customColorCount() . . . . .	8
4.1.2.6	fadeSpeed(uint8_t fadeSpeed) . . . . .	8
4.1.2.7	green(uint16_t i) . . . . .	8
4.1.2.8	mainColor() . . . . .	8
4.1.2.9	red(uint16_t i) . . . . .	8
4.1.2.10	setColor(uint16_t colorIndex, byte r, byte g, byte b) . . . . .	8
4.1.2.11	setCustomColorCount(uint8_t count) . . . . .	8
4.1.2.12	setMainColor(byte r, byte g, byte b) . . . . .	8

4.2	Single Color Routines	9
4.2.1	Detailed Description	9
4.2.2	Function Documentation	9
4.2.2.1	singleBlink(uint8_t red, uint8_t green, uint8_t blue)	9
4.2.2.2	singleFade(uint8_t red, uint8_t green, uint8_t blue, uint8_t fadeSpeed, boolean shouldUpdate)	9
4.2.2.3	singleGlimmer(uint8_t red, uint8_t green, uint8_t blue, long percent, boolean shouldUpdate)	10
4.2.2.4	singleSolid(uint8_t red, uint8_t green, uint8_t blue)	10
4.3	Multi Colors Routines	11
4.3.1	Detailed Description	11
4.3.2	Function Documentation	11
4.3.2.1	multiBarsMoving(ESColorGroup colorGroup, byte barSize)	11
4.3.2.2	multiBarsSolid(ESColorGroup colorGroup, byte barSize)	11
4.3.2.3	multiFade(ESColorGroup colorGroup)	11
4.3.2.4	multiGlimmer(ESColorGroup colorGroup, long percent)	12
4.3.2.5	multiRandomIndividual(ESColorGroup colorGroup)	12
4.3.2.6	multiRandomSolid(ESColorGroup colorGroup)	12
<b>5</b>	<b>Class Documentation</b>	<b>13</b>
5.1	RoutinesRGB Class Reference	13
5.1.1	Detailed Description	14
5.1.2	Constructor & Destructor Documentation	14
5.1.2.1	RoutinesRGB(uint16_t ledCount)	14
5.1.3	Member Function Documentation	15
5.1.3.1	resetToDefaults()	15
<b>6</b>	<b>File Documentation</b>	<b>17</b>
6.1	ColorPresets.h File Reference	17
6.1.1	Detailed Description	17
6.2	LightingProtocols.h File Reference	18
6.2.1	Detailed Description	18
6.2.2	Enumeration Type Documentation	18
6.2.2.1	ESColorGroup	18
6.2.2.2	ELightingRoutine	19
6.2.2.3	EPacketHeader	20
	<b>Index</b>	<b>21</b>

# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Getters and Setters . . . . .	7
Single Color Routines . . . . .	9
Multi Colors Routines . . . . .	11



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

#### [RoutinesRGB](#)

An Arduino library that provides a set of RGB lighting routines for compatible LED array hardware [13](#)





## Chapter 3

# File Index

### 3.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">ColorPresets.h</a>	17
<a href="#">LightingProtocols.h</a>	18
<b>RoutinesRGB.h</b>	<b>??</b>



# Chapter 4

## Module Documentation

### 4.1 Getters and Setters

#### Functions

- void [RoutinesRGB::setMainColor](#) (byte r, byte g, byte b)
- void [RoutinesRGB::setColor](#) (uint16\_t colorIndex, byte r, byte g, byte b)
- void [RoutinesRGB::setCustomColorCount](#) (uint8\_t count)
- uint8\_t [RoutinesRGB::customColorCount](#) ()
- void [RoutinesRGB::brightness](#) (uint8\_t brightness)
- void [RoutinesRGB::fadeSpeed](#) (uint8\_t fadeSpeed)
- void [RoutinesRGB::blinkSpeed](#) (uint8\_t blinkSpeed)
- Color [RoutinesRGB::mainColor](#) ()
- Color [RoutinesRGB::color](#) (uint16\_t i)
- uint8\_t [RoutinesRGB::red](#) (uint16\_t i)
- uint8\_t [RoutinesRGB::green](#) (uint16\_t i)
- uint8\_t [RoutinesRGB::blue](#) (uint16\_t i)

#### 4.1.1 Detailed Description

These are the getters and setters for [RoutinesRGB](#) that are used to control the settings and the colors.

#### 4.1.2 Function Documentation

##### 4.1.2.1 void [RoutinesRGB::blinkSpeed](#) ( uint8\_t *blinkSpeed* )

Sets how many updates to wait before changing the light state in the blink routine and in routines that switch between solid colors.

#### Parameters

<i>blinkSpeed</i>	a value between 1 and 255 representing how fast to blink. A value of 1 will make it blink on every frame, which may be too fast when used with other routines.
-------------------	--

#### 4.1.2.2 `uint8_t RoutinesRGB::blue ( uint16_t i )`

Retrieve the b value at a given index in the buffer.

#### 4.1.2.3 `void RoutinesRGB::brightness ( uint8_t brightness )`

Set the brightness between 0 and 100. 0 is off, 100 is full power.

#### 4.1.2.4 `RoutinesRGB::Color RoutinesRGB::color ( uint16_t i )`

Retrieve the color at the given index.

#### 4.1.2.5 `uint8_t RoutinesRGB::customColorCount ( )`

Retrieve the amount of colors that are used from the custom array.

#### 4.1.2.6 `void RoutinesRGB::fadeSpeed ( uint8_t fadeSpeed )`

Sets the speed of routines that fade between colors between 1 and 100. A fade speed of 1 is the slowest possible fade.

#### 4.1.2.7 `uint8_t RoutinesRGB::green ( uint16_t i )`

Retrieve the g value at a given index in the buffer.

#### 4.1.2.8 `RoutinesRGB::Color RoutinesRGB::mainColor ( )`

Retrieve the main color, which is used for single color routines.

#### 4.1.2.9 `uint8_t RoutinesRGB::red ( uint16_t i )`

Retrieve the r value at a given index in the buffer.

#### 4.1.2.10 `void RoutinesRGB::setColor ( uint16_t colorIndex, byte r, byte g, byte b )`

Set the color in the custom color array at the provided index. *colorIndex* must be less than the size of the custom color array or else it won't have any effect.

#### 4.1.2.11 `void RoutinesRGB::setCustomColorCount ( uint8_t count )`

Sets the amount of colors used in custom multi color routines. This is useful when you want to use a subset of the custom colors. The value given must be less than the size of the custom array or else it will be set to use the entire array.

#### 4.1.2.12 `void RoutinesRGB::setMainColor ( byte r, byte g, byte b )`

Sets the color used for single color routines.

## 4.2 Single Color Routines

### Functions

- void `RoutinesRGB::singleSolid` (uint8\_t red, uint8\_t green, uint8\_t blue)
- void `RoutinesRGB::singleBlink` (uint8\_t red, uint8\_t green, uint8\_t blue)
- void `RoutinesRGB::singleFade` (uint8\_t red, uint8\_t green, uint8\_t blue, uint8\_t fadeSpeed, boolean shouldUpdate)
- void `RoutinesRGB::singleGlimmer` (uint8\_t red, uint8\_t green, uint8\_t blue, long percent, boolean shouldUpdate)

### 4.2.1 Detailed Description

These routines each take a R, G, and B value as parameters to generate a color. This color is the only color used by the routine.

Blink, fade, and glimmer, should be called repeatedly on a loop for their full effect. The speed of the loop determines how fast the LEDs update.

### 4.2.2 Function Documentation

#### 4.2.2.1 void `RoutinesRGB::singleBlink` ( uint8\_t *red*, uint8\_t *green*, uint8\_t *blue* )

Switches between ON and OFF states using the provided color.

##### Parameters

<i>red</i>	strength of red LED, between 0 and 255
<i>green</i>	strength of green LED, between 0 and 255
<i>blue</i>	strength of blue LED, between 0 and 255

#### 4.2.2.2 void `RoutinesRGB::singleFade` ( uint8\_t *red*, uint8\_t *green*, uint8\_t *blue*, uint8\_t *fadeSpeed*, boolean *shouldUpdate* )

Fades the LEDs on and off based on the provided color. Uses the parameter fadeSpeed to determine how fast to fade. A larger number leads to a slower fade.

##### Parameters

<i>red</i>	strength of red LED, between 0 and 255
<i>green</i>	strength of green LED, between 0 and 255
<i>blue</i>	strength of blue LED, between 0 and 255
<i>fadeSpeed</i>	how many ticks it takes to fade. Higher numbers are slower.

4.2.2.3 `void RoutinesRGB::singleGlimmer ( uint8_t red, uint8_t green, uint8_t blue, long percent, boolean shouldUpdate )`

Set every LED to the provided color. A subset of the LEDs based on the percent parameter will be less bright than the rest of the LEDs.

**Parameters**

<i>red</i>	strength of red LED, between 0 and 255
<i>green</i>	strength of green LED, between 0 and 255
<i>blue</i>	strength of blue LED, between 0 and 255
<i>percent</i>	determines how many LEDs will be slightly dimmer than the rest

4.2.2.4 `void RoutinesRGB::singleSolid ( uint8_t red, uint8_t green, uint8_t blue )`

Set every LED to the provided color.

**Parameters**

<i>red</i>	strength of red LED, between 0 and 255
<i>green</i>	strength of green LED, between 0 and 255
<i>blue</i>	strength of blue LED, between 0 and 255

## 4.3 Multi Colors Routines

### Functions

- void `RoutinesRGB::multiGlimmer` (`EColorGroup` colorGroup, long percent)
- void `RoutinesRGB::multiFade` (`EColorGroup` colorGroup)
- void `RoutinesRGB::multiRandomIndividual` (`EColorGroup` colorGroup)
- void `RoutinesRGB::multiRandomSolid` (`EColorGroup` colorGroup)
- void `RoutinesRGB::multiBarsSolid` (`EColorGroup` colorGroup, byte barSize)
- void `RoutinesRGB::multiBarsMoving` (`EColorGroup` colorGroup, byte barSize)

#### 4.3.1 Detailed Description

These routines use multiple colors. They all take the parameter of `colorGroup` which is used to determine which set of colors to use. The custom color array is `eCustom`, all other values for `colorGroup` come from presets color. Go to the project's github for a full list of the colorGroups and their corresponding values.

All routines except `multiBarsSolid` should be called repeatedly on a loop for their full effect. The speed of the loop determines how fast the LEDs update.

#### 4.3.2 Function Documentation

##### 4.3.2.1 void `RoutinesRGB::multiBarsMoving` ( `EColorGroup` colorGroup, byte barSize )

Provides a similar effect as `multiBarSolid`, but the alternating patches move up one LED index on each frame update to create a "scrolling" effect.

##### Parameters

<i>colorGroup</i>	the color group to use for the routine. <code>eCustom</code> is the custom array, all other values are preset groups.
<i>barSize</i>	how many LEDs before switching to the other bar.

##### 4.3.2.2 void `RoutinesRGB::multiBarsSolid` ( `EColorGroup` colorGroup, byte barSize )

Uses the chosen color group to set the LEDs in alternating patches with a size of `barSize`.

##### Parameters

<i>colorGroup</i>	the color group to use for the routine. <code>eCustom</code> is the custom array, all other values are preset groups.
<i>barSize</i>	how many LEDs before switching to the other bar.

##### 4.3.2.3 void `RoutinesRGB::multiFade` ( `EColorGroup` colorGroup )

Fades between all the colors used by the color group.

## Parameters

<i>colorGroup</i>	the color group to use for the routine. eCustom is the custom array, all other values are preset groups.
-------------------	--

4.3.2.4 void RoutinesRGB::multiGlimmer ( EColorGroup *colorGroup*, long *percent* )

This method uses its percent parameter to dim LEDs randomly, similar to the standard glimmer mode. It also uses the percent to randomly change the color of select LEDs to a color in the chosen color group. The base color is the first from the chosen color group.

## Parameters

<i>colorGroup</i>	the color group to use for the routine. eCustom is the custom array, all other values are preset groups.
<i>percent</i>	percent of LEDs that will get the glimmer applied

4.3.2.5 void RoutinesRGB::multiRandomIndividual ( EColorGroup *colorGroup* )

sets each individual LED as a random color from the chosen color group.

## Parameters

<i>colorGroup</i>	the color array to use for the routine. eCustom is the custom array, all other values are colorGroup arrays
-------------------	---

4.3.2.6 void RoutinesRGB::multiRandomSolid ( EColorGroup *colorGroup* )

A random color is chosen from the chosen color group and applied to each LED.

## Parameters

<i>colorGroup</i>	the color group to use for the routine. eCustom is the custom array, all other values are preset groups.
-------------------	--



## Chapter 5

# Class Documentation

### 5.1 RoutinesRGB Class Reference

An Arduino library that provides a set of RGB lighting routines for compatible LED array hardware.

```
#include <RoutinesRGB.h>
```

#### Public Member Functions

- [RoutinesRGB](#) (uint16\_t ledCount)
- void [resetToDefaults](#) ()
- void [setMainColor](#) (byte r, byte g, byte b)
- void [setColor](#) (uint16\_t colorIndex, byte r, byte g, byte b)
- void [setCustomColorCount](#) (uint8\_t count)
- uint8\_t [customColorCount](#) ()
- void [brightness](#) (uint8\_t brightness)
- void [fadeSpeed](#) (uint8\_t fadeSpeed)
- void [blinkSpeed](#) (uint8\_t blinkSpeed)
- Color [mainColor](#) ()
- Color [color](#) (uint16\_t i)
- uint8\_t [red](#) (uint16\_t i)
- uint8\_t [green](#) (uint16\_t i)
- uint8\_t [blue](#) (uint16\_t i)
- void [singleSolid](#) (uint8\_t [red](#), uint8\_t [green](#), uint8\_t [blue](#))
- void [singleBlink](#) (uint8\_t [red](#), uint8\_t [green](#), uint8\_t [blue](#))
- void [singleFade](#) (uint8\_t [red](#), uint8\_t [green](#), uint8\_t [blue](#), uint8\_t [fadeSpeed](#), boolean shouldUpdate)
- void [singleGlimmer](#) (uint8\_t [red](#), uint8\_t [green](#), uint8\_t [blue](#), long percent, boolean shouldUpdate)
- void [multiGlimmer](#) ([EColorGroup](#) colorGroup, long percent)
- void [multiFade](#) ([EColorGroup](#) colorGroup)
- void [multiRandomIndividual](#) ([EColorGroup](#) colorGroup)
- void [multiRandomSolid](#) ([EColorGroup](#) colorGroup)
- void [multiBarsSolid](#) ([EColorGroup](#) colorGroup, byte barSize)
- void [multiBarsMoving](#) ([EColorGroup](#) colorGroup, byte barSize)

### 5.1.1 Detailed Description

An Arduino library that provides a set of RGB lighting routines for compatible LED array hardware.

#### Version

v1.9.2

#### Date

May 5, 2016

#### Author

Tim Seemann

#### Copyright

MIT License

This library has been tested with SeeedStudio Rainbowduinos, quite a few of the Adafruit Neopixels products, and a standard RGB LED. Sample code is provided in the git repo for all tested hardware in the samples folder of the git repository.

If you are starting a project from scratch, first you'll need to make a global object in the arduino sketch:

```
RoutinesRGB routines = RoutinesRGB(LED_COUNT);
```

where `LED_COUNT` is the number of LEDs in your array.

The library produces lighting routines based on the functions used and stores the routine in its internal buffers. These buffers can then be accessed by getters and displayed on the LED hardware. For routines that change over time, this processed should be repeated on a loop. For example, here is how you would make a red blinking light with the library and a Neopixels board:

First, call this function to store the routine in the library's internal buffers:

```
routines.singleBlink(255, 0, 0);
```

Then, update the LED array with the values from the library's RGB buffer. The way to do this will vary from hardware to hardware, but for a NeoPixels sample, it would look something like this:

```
for (int x = 0; x < LED_COUNT; x++) {  
    pixels.setPixelColor(x, pixels.Color(routines.red(x),  
                                         routines.green(x),  
                                         routines.blue(x)));  
}  
pixels.show();
```

By this point, the LEDs should be showing red. To achieve the blink effect, put both of these in your `loop()` function and then put a delay between updates. This delay will be used to determine how fast the LED's blink.

### 5.1.2 Constructor & Destructor Documentation

#### 5.1.2.1 RoutinesRGB::RoutinesRGB ( uint16\_t ledCount )

Required constructor. The library should be stored in global memory and allocated only once at startup.

It will allocate `4 * ledCount` bytes.

## Parameters

<i>ledCount</i>	number of individual RGB LEDs.
-----------------	--------------------------------

### 5.1.3 Member Function Documentation

#### 5.1.3.1 void RoutinesRGB::resetToDefaults ( )

Resets all internal values to the original values.



## Chapter 6

# File Documentation

### 6.1 ColorPresets.h File Reference

```
#include <avr/pgmspace.h>
```

#### 6.1.1 Detailed Description

##### Version

v1.9.2

##### Date

May 5, 2016

##### Author

Tim Seemann

##### Copyright

MIT License

These color presets are stored in program memory and loaded into a buffer when accessed. This makes the presets read-only, but in return, it allows them to take a much smaller hit on SRAM usage.

## 6.2 LightingProtocols.h File Reference

### Enumerations

#### 6.2.1 Detailed Description

##### Version

v1.9.2

##### Date

May 5, 2016

##### Author

Tim Seemann

##### Copyright

MIT License

This file defines the protocols used for the Arduino libraries and the GUI.

A slightly modified version of this file exists in the Qt GUI project. None of the modifications change the naming, documentation, or order of the protocols. Instead, the changes allow the GUI version to use the strongly typed enums that were made available in C++11.

#### 6.2.2 Enumeration Type Documentation

##### 6.2.2.1 enum EColorGroup

used during multi color routines to determine which colors to use in the routine. eCustom uses the custom color array, eAll generates its colors randomly. All other values use presets based around overall themes.

##### Enumerator

###### **eCustom 0**

*Use the custom color array instead of a preset group.*

###### **eWater 1**

*Shades of blue with some teal.*

###### **eFrozen 2**

*Shades of teal with some blue, white, and light purple.*

###### **eSnow 3**

*Shades of white with some blue and teal.*

###### **eCool 4**

*Based on the cool colors: blue, green, and purple.*

###### **eWarm 5**

*Based on the warm colors: red, orange, and yellow.*

**eFire 6**

*Similar to the warm set, but with an emphasis on oranges to give it a fire-like glow.*

**eEvil 7**

*Mostly red, with some other, evil highlights.*

**eCorrosive 8**

*Greens and whites, similar to radioactive goo from a 90s kids cartoon.*

**ePoison 9**

*A purple-based theme. Similar to poison vials from a 90s kids cartoon.*

**eRose 10**

*Shades of pink, red, and white.*

**ePinkGreen 11**

*The colors of watermelon candy. bright pinks and bright green.*

**eRedWhiteBlue 12**

*Bruce Springsteen's favorite color scheme, good ol' red, white, and blue.*

**eRGB 13**

*red, green, and blue.*

**eCMY 14**

*Cyan, magenta, yellow.*

**eSixColor 15**

*Red, yellow, green, cyan, blue, magenta.*

**eSevenColor 16**

*Red, yellow, green, cyan, blue, magenta, white.*

**eAll 17**

*Rather than using using preset colors, it uses all possible colors.*

## 6.2.2.2 enum ELightingRoutine

Each routine makes the LEDs shine in different ways. There are two main types of routines: Single Color Routines use a single color while Multi Color Routines rely on an EColorGroup.

## Enumerator

**eOff 0**

*Turns off the LEDs.*

**eSingleSolid 1**

*Shows a single color at a fixed brightness.*

**eSingleBlink 2**

*Alternates between showing a single color at a fixed brightness and turning the LEDs completely off.*

**eSingleFade 3**

*Linear fade of the brightness of the LEDs.*

**eSingleGlimmer 4**

*Randomly dims some of the LEDs to give a glimmer effect.*

**eMultiGlimmer 5**

*Uses the first color of the array as the base color and uses the other colors for a glimmer effect.*

**eMultiFade 6**

*Fades slowly between each color in the array.*

**eMultiRandomSolid 7**

*Chooses a random color from the array and lights all all LEDs to match that color.*

**eMultiRandomIndividual 8**

*Chooses a random color from the array for each individual LED.*

**eMultiBarsSolid 9**

*Draws the colors of the array in alternating groups of equal size.*

**eMultiBarsMoving 10**

*Draws the colors of the array in alternating groups of equal size. On each update, it moves those groups one index to the right, creating a scrolling effect.*

**6.2.2.3 enum EPacketHeader**

Message headers for packets coming over serial.

Enumerator

**eModeChange 0**

*Takes one int parameter that gets cast to ELightingMode.*

**eMainColorChange 1**

*Takes 3 parameters, a 0-255 representation of Red, Green, and Blue.*

**eCustomArrayColorChange 2**

*Takes four parameters, three parameters, the LED, a 0-255 representation of Red, Green, and Blue.*

**eBrightnessChange 3**

*Takes one parameter, sets the brightness between 0 and 100.*

**eSpeedChange 4**

*Takes one parameter, sets the delay value 1 - 23767.*

**eCustomColorCountChange 5**

*Change the number of colors used in a custom array routine.*

**eIdleTimeoutChange 6**

*Set to 0 to turn off, set to any other number minutes until idle timeout happens.*

**eResetSettingsToDefaults 7**

*Resets all values inside of [RoutinesRGB](#) back to their default values. Useful for soft resetting the LED hardware.*



# Index

blinkSpeed  
    Getters and Setters, [7](#)  
blue  
    Getters and Setters, [7](#)  
brightness  
    Getters and Setters, [8](#)  
  
color  
    Getters and Setters, [8](#)  
ColorPresets.h, [17](#)  
customColorCount  
    Getters and Setters, [8](#)  
  
eAll  
    LightingProtocols.h, [19](#)  
eBrightnessChange  
    LightingProtocols.h, [20](#)  
eCMY  
    LightingProtocols.h, [19](#)  
EColorGroup  
    LightingProtocols.h, [18](#)  
eCool  
    LightingProtocols.h, [18](#)  
eCorrosive  
    LightingProtocols.h, [19](#)  
eCustom  
    LightingProtocols.h, [18](#)  
eCustomArrayColorChange  
    LightingProtocols.h, [20](#)  
eCustomColorCountChange  
    LightingProtocols.h, [20](#)  
eEvil  
    LightingProtocols.h, [19](#)  
eFire  
    LightingProtocols.h, [18](#)  
eFrozen  
    LightingProtocols.h, [18](#)  
eIdleTimeoutChange  
    LightingProtocols.h, [20](#)  
ELightingRoutine  
    LightingProtocols.h, [19](#)  
eMainColorChange  
    LightingProtocols.h, [20](#)  
eModeChange  
    LightingProtocols.h, [20](#)  
eMultiBarsMoving  
    LightingProtocols.h, [20](#)  
eMultiBarsSolid  
    LightingProtocols.h, [19](#)  
eMultiFade

    LightingProtocols.h, [19](#)  
eMultiGlimmer  
    LightingProtocols.h, [19](#)  
eMultiRandomIndividual  
    LightingProtocols.h, [19](#)  
eMultiRandomSolid  
    LightingProtocols.h, [19](#)  
eOff  
    LightingProtocols.h, [19](#)  
EPacketHeader  
    LightingProtocols.h, [20](#)  
ePinkGreen  
    LightingProtocols.h, [19](#)  
ePoison  
    LightingProtocols.h, [19](#)  
eRGB  
    LightingProtocols.h, [19](#)  
eRedWhiteBlue  
    LightingProtocols.h, [19](#)  
eResetSettingsToDefaults  
    LightingProtocols.h, [20](#)  
eRose  
    LightingProtocols.h, [19](#)  
eSevenColor  
    LightingProtocols.h, [19](#)  
eSingleBlink  
    LightingProtocols.h, [19](#)  
eSingleFade  
    LightingProtocols.h, [19](#)  
eSingleGlimmer  
    LightingProtocols.h, [19](#)  
eSingleSolid  
    LightingProtocols.h, [19](#)  
eSixColor  
    LightingProtocols.h, [19](#)  
eSnow  
    LightingProtocols.h, [18](#)  
eSpeedChange  
    LightingProtocols.h, [20](#)  
eWarm  
    LightingProtocols.h, [18](#)  
eWater  
    LightingProtocols.h, [18](#)  
  
fadeSpeed  
    Getters and Setters, [8](#)  
  
Getters and Setters, [7](#)  
    blinkSpeed, [7](#)  
    blue, [7](#)

- brightness, 8
- color, 8
- customColorCount, 8
- fadeSpeed, 8
- green, 8
- mainColor, 8
- red, 8
- setColor, 8
- setCustomColorCount, 8
- setMainColor, 8
- green
  - Getters and Setters, 8
- LightingProtocols.h, 18
  - eAll, 19
  - eBrightnessChange, 20
  - eCMY, 19
  - EColorGroup, 18
  - eCool, 18
  - eCorrosive, 19
  - eCustom, 18
  - eCustomArrayColorChange, 20
  - eCustomColorCountChange, 20
  - eEvil, 19
  - eFire, 18
  - eFrozen, 18
  - eIdleTimeoutChange, 20
  - ELightingRoutine, 19
  - eMainColorChange, 20
  - eModeChange, 20
  - eMultiBarsMoving, 20
  - eMultiBarsSolid, 19
  - eMultiFade, 19
  - eMultiGlimmer, 19
  - eMultiRandomIndividual, 19
  - eMultiRandomSolid, 19
  - eOff, 19
  - EPacketHeader, 20
  - ePinkGreen, 19
  - ePoison, 19
  - eRGB, 19
  - eRedWhiteBlue, 19
  - eResetSettingsToDefaults, 20
  - eRose, 19
  - eSevenColor, 19
  - eSingleBlink, 19
  - eSingleFade, 19
  - eSingleGlimmer, 19
  - eSingleSolid, 19
  - eSixColor, 19
  - eSnow, 18
  - eSpeedChange, 20
  - eWarm, 18
  - eWater, 18
- mainColor
  - Getters and Setters, 8
- Multi Colors Routines, 11
  - multiBarsMoving, 11
  - multiBarsSolid, 11
  - multiFade, 11
  - multiGlimmer, 12
  - multiRandomIndividual, 12
  - multiRandomSolid, 12
- red
  - Getters and Setters, 8
- resetToDefaults
  - RoutinesRGB, 15
- RoutinesRGB, 13
  - resetToDefaults, 15
  - RoutinesRGB, 14
- setColor
  - Getters and Setters, 8
- setCustomColorCount
  - Getters and Setters, 8
- setMainColor
  - Getters and Setters, 8
- Single Color Routines, 9
  - singleBlink, 9
  - singleFade, 9
  - singleGlimmer, 9
  - singleSolid, 10
- singleBlink
  - Single Color Routines, 9
- singleFade
  - Single Color Routines, 9
- singleGlimmer
  - Single Color Routines, 9
- singleSolid
  - Single Color Routines, 10