

---

# **Pigment**

***Release 0.5.0***

**Ben Soyka**

**Jan 05, 2021**



# CONTENTS

<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Documentation Contents</b>	<b>5</b>
2.1	Reference . . . . .	5
2.2	Exceptions . . . . .	7
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



**Pigment** is a set of Python utilities for colors.

```
>>> red = Color(255, 0, 0)
>>> blue = Color(0, 0, 255)
>>> blend(red, blue).hsv
(300, 100, 50)
```



## INSTALLATION

Pigment is available on PyPI:

```
$ python -m pip install pigment
```

Pigment officially supports Python 3.5+.





## DOCUMENTATION CONTENTS

### 2.1 Reference

Python utilities for colors

**class** `pigment.Color` (*red: int, green: int, blue: int*)  
Represents a color

**Parameters**

- **red** (*int*) – The color’s red RGB component (0-255)
- **green** (*int*) – The color’s green RGB component (0-255)
- **blue** (*int*) – The color’s blue RGB component (0-255)

---

**Note:** The *cmypk*, *hex\_code*, *hls*, *hsv*, and *rgb* properties can be set to update the color object.

---

**property** `cmypk`

The color as a CMYK tuple

- Cyan (0-100)
- Magenta (0-100)
- Yellow (0-100)
- Key/Black (0-100)

**classmethod** `from_css_name` (*css\_color: str*)

Gets a color from a CSS color name

**Parameters** `css_color` (*str*) – The name of the CSS color

**Returns** *Color*

**property** `hex_code`

The color’s hex code

**property** `hls`

The color as an HLS tuple

- Hue: color (0-360)
- Lightness: amount of white vs. color (0-100)
- Saturation: amount of gray vs. color (0-100)

**property** `hsv`

The color as an HSV tuple

- Hue: color (0-360)
- Saturation: amount of gray vs. color (0-100)
- Value: amount of black vs. color (0-100)

**property** `hue`

The color's hue (0-360)

**classmethod** `random` (*red: tuple = (0, 255), green: tuple = (0, 255), blue: tuple = (0, 255)*)

Generates a random color

This works by generating random red, green, and blue values using `random.randint()` from the standard library using the min/max values specified if any

**Parameters**

- **red** (*tuple*) – The two arguments to pass for the red value
- **green** (*tuple*) – The two arguments to pass for the green value
- **blue** (*tuple*) – The two arguments to pass for the blue value

**Returns** *Color*

**property** `rgb`

The color as an RGB tuple

- Red (0-255)
- Green (0-255)
- Blue (0-255)

`pigment.blend` (*color1: pigment.Color, color2: pigment.Color*) → *pigment.Color*

Blends two colors together

**Parameters**

- **color1** (*Color*) – The first color
- **color2** (*Color*) – The second color

**Returns** *Color*

`pigment.normalize_hex` (*hex\_code: str*) → *str*

Normalizes a hex color code

Removes the leading # if there is one, expands 3-character hex codes, and lowercases the hex code

**Parameters** **hex\_code** (*str*) – A hex code to normalize

**Returns** The normalized hex code

**Return type** *str*

**Raises** *WrongLengthError* – The provided hex code had the wrong length

## 2.2 Exceptions

**exception** `pigment.exceptions.PigmentError`

The base class for all Pigment exceptions

---

**exception** `pigment.exceptions.InvalidRGBValue`

A value was invalid when converting or saving a color in RGB form

---

**Note:** Also a subclass of `ValueError`

---

**exception** `pigment.exceptions.WrongLengthError` (*argument: Optional[str] = None*)

The argument provided had an invalid length

---

**Note:** Also a subclass of `ValueError`

---

**argument**

The name of the argument

**Type** `str`



## PYTHON MODULE INDEX

### p

pigment, [5](#)

pigment.exceptions, [7](#)



## INDEX

### A

`argument` (*pigment.exceptions.WrongLengthError* attribute), 7

### B

`blend()` (in module *pigment*), 6

### C

`cmypk()` (*pigment.Color* property), 5

`Color` (class in *pigment*), 5

### F

`from_css_name()` (*pigment.Color* class method), 5

### H

`hex_code()` (*pigment.Color* property), 5

`hls()` (*pigment.Color* property), 5

`hsv()` (*pigment.Color* property), 5

`hue()` (*pigment.Color* property), 6

### I

`InvalidRGBValue`, 7

### M

module

*pigment*, 5

*pigment.exceptions*, 7

### N

`normalize_hex()` (in module *pigment*), 6

### P

*pigment*

    module, 5

*pigment.exceptions*

    module, 7

`PigmentError`, 7

### R

`random()` (*pigment.Color* class method), 6

`rgb()` (*pigment.Color* property), 6

### W

`WrongLengthError`, 7