

# CSS- 09- j- Modules de couleur

Sass offre différents modules maison (*built-in*) permettant de simplifier l'écriture de code CSS. Parmi ceux-ci, le plus populaire est le module de couleurs que nous examinerons aujourd'hui.

- [built-in modules](#)

## Importation

Afin d'éviter d'affecter négativement la performance du code CSS, les modules ne sont pas importés par défaut. Il faut donc penser à les importer si l'on souhaite les utiliser.

Par exemple, afin d'importer le module de couleurs, il faut écrire:

```
@use "sass:color";
```

Suite à cet import, plusieurs fonctionnalités de traitement de couleurs s'ajouteront à Sass.

## Whiteness & Blackness

Les fonctionnalités *whiteness* et *blackness* permettent d'altérer une couleur en lui ajoutant du blanc ○ ou du noir ●. Le résultat correspond au résultat obtenu si un galon de peinture verte était mélangé à moitié (50%) avec un galon de peinture blanche ou à moitié (50%) avec un galon de peinture noire.

Les unités doivent être exprimées en pourcentage (%) pouvant-être positif ou négatif.

## Exemple

Code HTML :

```
<div class="element whiteness">
  <div class="title">Whiteness</div>
  <div class="module">color.adjust($color, $whiteness: 50%)</div>
</div>

<div class="element original">Original</div>

<div class="element blackness">
  <div class="title">Blackness</div>
  <div class="module">color.adjust($color, $blackness: 50%)</div>
</div>
```

Code SCSS :

```
@use "sass:color"; // 📁 Ne pas oublier
$color: #00c774;

.original { background-color: $color; }

.whiteness {
  background: color.adjust($color, $whiteness: 50%);
}

.blackness {
  background: color.adjust($color, $blackness: 50%);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
  margin: -2px 0;
  padding: 0;
  overflow: hidden;
}

.element {
  display: flex;
  flex-direction: column;
  width: 100%;
  color: #fff;
  flex-grow: 1;
  margin: 2px 0;
  padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }
```

CSS 09 j

Réalisation : Guillaume DELACROIX Formateur AFPA

12 janvier 2023

**Afpa** 

```
@media (min-width: 992px) {  
  body { font-size: 19px; }  
}
```

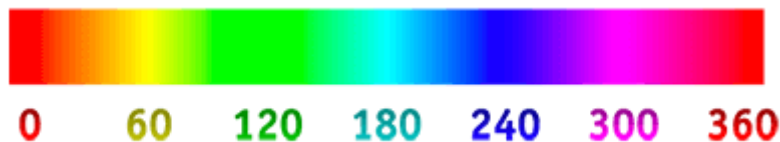
Rendu :



## Hue

La fonctionnalité hue permet de changer la teinte d'une couleur toute en gardant exactement la même saturation et la même luminosité.

Les unités données doivent-être des nombres positifs ou négatifs déplaçant la position de la teinte actuelle sur l'échelle de teintes.



## Exemple

### Code HTML

```
<div class="element hue1">
  <div class="title">Hue 100</div>
  <div class="module">color.adjust($color, $hue: 100)</div>
</div>

<div class="element original">Original</div>

<div class="element hue2">
  <div class="title">Hue -100</div>
  <div class="module">color.adjust($color, $hue: -100)</div>
</div>
```

### Code SCSS

```
@use "sass:color"; // Ne pas oublier
$color: #00c774;

.original { background-color: $color; }

.hue1 {
  background: color.adjust($color, $hue: 100);
}

.hue2 {
  background: color.adjust($color, $hue: -100);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
}
```

```

margin: -2px 0;
padding: 0;
overflow: hidden;
}

.element {
display: flex;
flex-direction: column;
width: 100%;
color: #fff;
flex-grow: 1;
margin: 2px 0;
padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }

@media (min-width: 992px) {
body { font-size: 19px; }
}


```

## Rendu




La teinte actuelle à une valeur de 155.

En haut , elle est incrémentée de 100, donc atteint à 255.

En bas , elle est diminuée de 100, donc atteint 55.

## Saturation & Grayscale

La fonctionnalité saturation permet d'altérer l'intensité d'une couleur. Par exemple, en donnant une saturation de -50%, une couleur devient automatiquement 50% moins vibrante. La fonctionnalité `color.grayscale` quant à elle est l'équivalent d'une couleur ayant perdu toute son intensité, pour ainsi dire l'équivalent de lorsque cette couleur est imprimée en noir et blanc .

Les unités doivent être exprimées en pourcentage (%) pouvant-être positif ou négatif.

### Exemple

#### Code HTML

```
<div class="element saturation">
  <div class="title">Saturation</div>
  <div class="module">color.adjust($color, $saturation: -50%)</div>
</div>

<div class="element original">Original</div>

<div class="element grayscale">
  <div class="title">Grayscale</div>
  <div class="module">color.grayscale($color)</div>
</div>
```

#### Code SCSS

```
@use "sass:color"; // 🖨 Ne pas oublier
$color: #00c774;

.original { background-color: $color; }

.saturation {
  background: color.adjust($color, $saturation: -50%);
}

.grayscale {
  background: color.grayscale($color);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
  margin: -2px 0;
  padding: 0;
```

```

overflow: hidden;
}

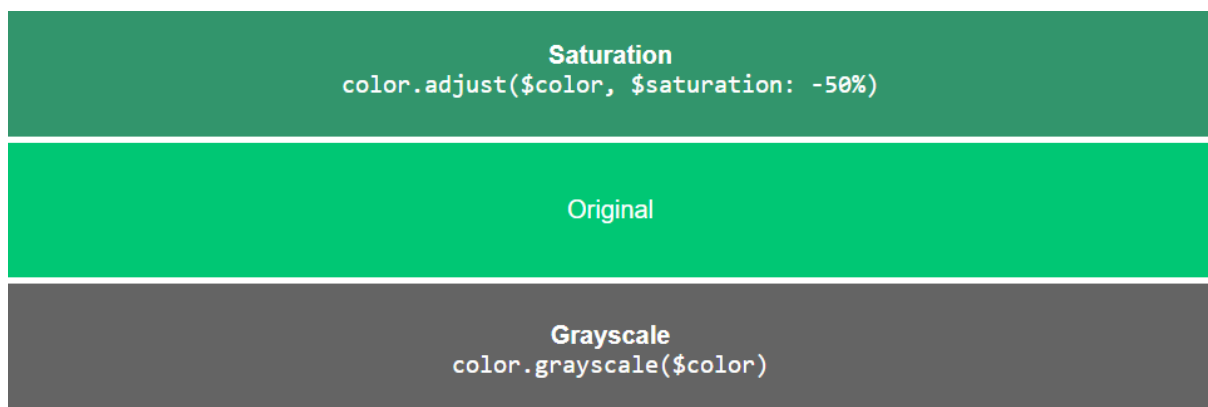
.element {
display: flex;
flex-direction: column;
width: 100%;
color: #fff;
flex-grow: 1;
margin: 2px 0;
padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }

@media (min-width: 992px) {
body { font-size: 19px; }
}

```

## Rendu



`color.adjust($color, $saturation: -100%) = color.grayscale($color)`,  
soit une couleur complètement désaturée.

## Lightness

La fonctionnalité lightness permet d'altérer la luminosité d'une couleur. Il est donc possible de la rendre plus lumineuse ou plus sombre via celle-ci.

Les unités doivent être exprimées en pourcentage (%) pouvant-être positif ou négatif.

## Exemple

Code HTML :

```
<div class="element positive">
  <div class="title">Lightness +</div>
  <div class="module">color.adjust($color, $lightness: 20%)</div>
</div>

<div class="element original">Original</div>

<div class="element negative">
  <div class="title">Lightness -</div>
  <div class="module">color.adjust($color, $lightness: -20%)</div>
</div>
```

Code SCSS :

```
@use "sass:color"; // 📄 Ne pas oublier
$color: #00c774;

.original { background-color: $color; }

.positive {
  background: color.adjust($color, $lightness: 20%);
}

.negative {
  background: color.adjust($color, $lightness: -20%);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
  margin: 0 -2px;
  padding: 0;
  overflow: hidden;
}

.element {
  display: flex;
  flex-direction: column;
  width: 100%;
```

CSS 09 j

Réalisation : Guillaume DELACROIX Formateur AFPA

12 janvier 2023

**Afpa** 



```
color: #fff;
flex-grow: 1;
margin: 2px 0;
padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }

@media (min-width: 992px) {
  body { font-size: 19px; }
}
```

Rendu :

**Lightness +**  
`color.adjust($color, $lightness: 20%)`

Original

**Lightness -**  
`color.adjust($color, $lightness: -20%)`

## Complement & Invert

La fonctionnalité `color.complement` permet d'obtenir la couleur complémentaire sur le cercle chromatique à la couleur passée en argument. Tandis que la fonctionnalité `color.invert` soustrait les valeurs de rouge, vert et bleu à 255 afin d'obtenir la couleur opposée exacte.

Certes, ces deux fonctionnalités se ressemblent beaucoup, mais il est parfois pratique d'avoir la flexibilité de choisir entre les deux.

### Exemple

Code HTML :

```
<div class="element complement">
  <div class="title">Complement</div>
  <div class="module">color.complement($color)</div>
</div>

<div class="element original">Original</div>

<div class="element invert">
  <div class="title">Invert</div>
  <div class="module">color.invert($color)</div>
</div>
```

Code SCSS :

```
@use "sass:color"; // Ne pas oublier
$color: #00c774;

.original { background-color: $color; }

.complement {
  background: color.complement($color);
}

.invert {
  background: color.invert($color);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
  margin: -2px 0;
  padding: 0;
  overflow: hidden;
}
```

```

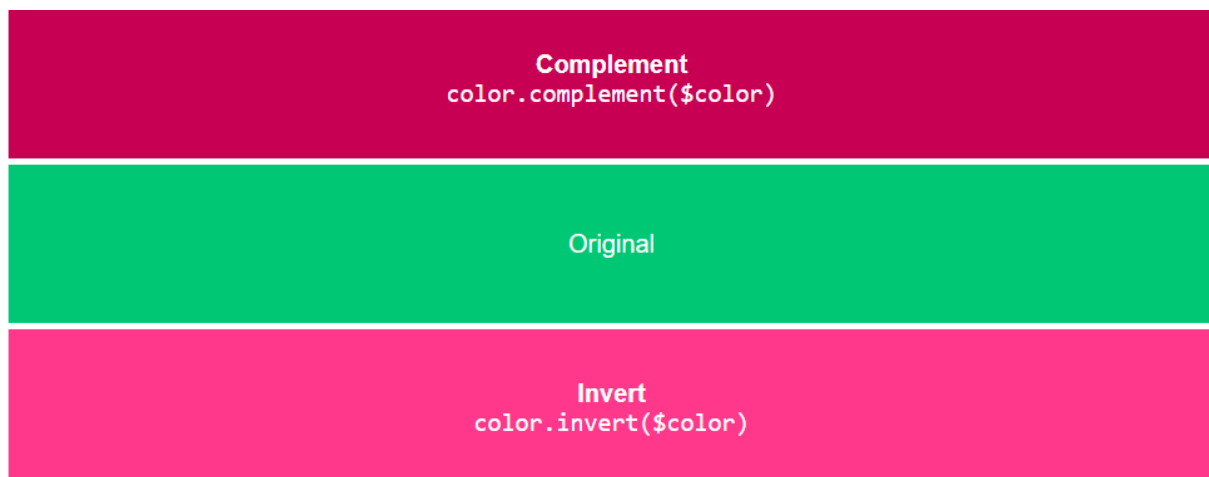
.element {
  display: flex;
  flex-direction: column;
  width: 100%;
  color: #fff;
  flex-grow: 1;
  margin: 2px 0;
  padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }

@media (min-width: 992px) {
  body { font-size: 19px; }
}

```

Rendu :



## Red, Green & Blue

Les fonctionnalités red, green et blue permettent d'augmenter ou de réduire le taux de rouge , vert  ou bleu  dans une couleur donnée.

Les unités données doivent-être des nombres positifs ou négatifs faisant varier le taux de rouge, vert ou bleu. Ces taux ont un minimum de 0 et un maximum de 255 qu'ils ne peuvent jamais dépasser.

## Exemple

Code HTML :

```
<div class="element red">
  <div class="title">Red</div>
  <div class="module">color.adjust($color, $red: 50)</div>
</div>

<div class="element green">
  <div class="title">Green</div>
  <div class="module">color.adjust($color, $green: 50)</div>
</div>

<div class="element blue">
  <div class="title">Blue</div>
  <div class="module">color.adjust($color, $blue: 50)</div>
</div>
```

Code SCSS :

```
@use "sass:color"; // 📌 Ne pas oublier
$color: #00c774;

.red {
  background: color.adjust($color, $red: 50);
}

.green {
  background: color.adjust($color, $green: 50);
}

.blue {
  background: color.adjust($color, $blue: 50);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
}
```

```

margin: -2px 0;
padding: 0;
overflow: hidden;
}

.element {
display: flex;
flex-direction: column;
width: 100%;
color: #fff;
flex-grow: 1;
margin: 2px 0;
padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }

@media (min-width: 992px) {
body { font-size: 19px; }
}

```

Rendu :



## Alpha

La fonctionnalité alpha permet de faire fluctuer le taux de transparence d'une couleur. Pour ce faire, il faut passer un nombre entre -1 et 1 s'ajoutant à la valeur courante d'alpha.

Par exemple, si une couleur est actuellement opaque (*alpha de 1*), il faudra lui donner la valeur -0.5 pour diminuer son opacité de 50%.

## Exemple

### Code HTML

```
<div class="element original">
  <div class="title">Original</div>
</div>

<div class="element alpha">
  <div class="title">Alpha</div>
  <div class="module">color.adjust($color, $alpha: 0.5)</div>
</div>
```

### Code SCSS

```
@use "sass:color"; // Ne pas oublier
$color: #00c774;

.original { background: $color; }

.alpha {
  background: color.adjust($color, $alpha: -0.5);
}

/* Layout */

body {
  flex-direction: column;
  font-size: min(16px, 3vw);
  background: url(https://assets.codepen.io/1189183/photoshop-tile.png);
  margin: -2px 0;
  padding: 0;
  overflow: hidden;
}

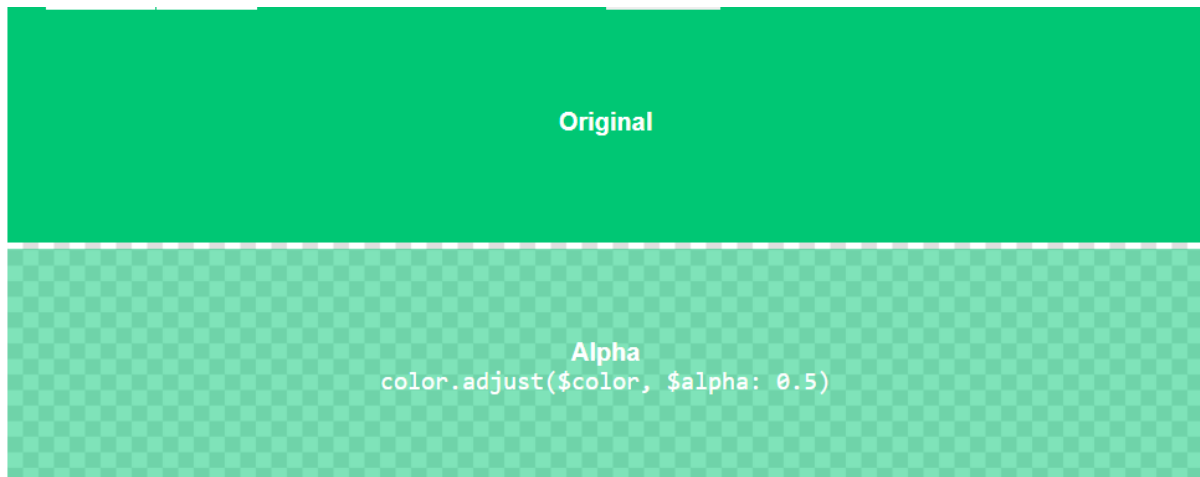
.element {
  display: flex;
  flex-direction: column;
  width: 100%;
  color: #fff;
  flex-grow: 1;
```

```
margin: 2px 0;
padding: 0.75em;
}

.title { font-weight: bold; }
.module { font-family: monospace; }


@media (min-width: 992px) {
  body { font-size: 19px; }
}
```

## Rendu



- [module color](#)

## Cheatsheet

Voici coup sur coup la même couleur verte  (\$color: #00c774;) altérée via des fonctionnalités du module de couleurs Sass:

<b>Whiteness</b> <code>color.adjust(\$color, \$whiteness: 20%)</code>
<b>Blackness</b> <code>color.adjust(\$color, \$blackness: 20%)</code>
<b>Hue</b> <code>color.adjust(\$color, \$hue: 100)</code>
<b>Saturation</b> <code>color.adjust(\$color, \$saturation: -50%)</code>
<b>Lightness</b> <code>color.adjust(\$color, \$lightness: 20%)</code>
<b>Grayscale</b> <code>color.grayscale(\$color)</code>
<b>Compelement</b> <code>color.complement(\$color)</code>
<b>Invert</b> <code>color.invert(\$color)</code>
<b>Red</b> <code>color.adjust(\$color, \$red: 50)</code>
<b>Green</b> <code>color.adjust(\$color, \$green: 50)</code>
<b>Blue</b> <code>color.adjust(\$color, \$Blue: 50)</code>
<b>Alpha</b> <code>color.adjust(\$color, \$alpha: -0.5)</code>