




CSS3 Módulo para cores

Publicado em: 2009-11-10

Atualizações: 2011-03-27 — 2015-08-17

Revisão: 2020-04-04

Introdução

O Módulo das CSS3 denominado [CSS3 Color Module - Level 3](#)  destina-se a especificar as diferentes maneiras de se atribuir valores para as propriedades CSS que admitem cores tais como `background-color` e `color`. O Módulo é uma Recomendação do W3C, e sua implementação é realidade nos navegadores modernos e nos IE9+

O objetivo deste módulo segundo as especificações do W3C é o seguinte:

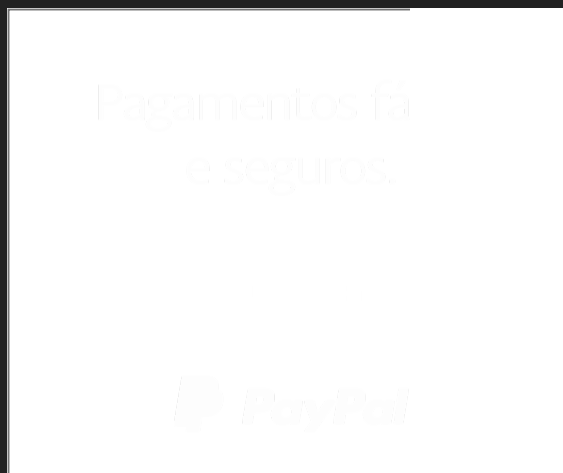
“

CSS (Cascading Style Sheets) é uma linguagem que descreve como devem ser apresentados os documentos HTML e XML na tela, no papel, em um dispositivo aural, etc. A language utiliza propriedades e valores para cores de textos, backgrounds, bordas e outras partes dos elementos em um documento. Esta especificação descreve os valores de cores e suas propriedades bem como opacidade de cores. Estão aqui incluídas as propriedades das CSS nível 2 e alguns novos valores para estas propriedades.

Declarando cores RGB, RGBA, HSL e HSLA

Nesse tutorial veremos a sintaxe e efeitos da declaração de cores com uso das funções CSS

`rgb()`, `rgba()`, `hsl()`, `hsla()`.



[topo](#)

Declaração de cores com uso da função CSS `rgb()`

`rgb` é abreviatura para:

r = red (vermelha)

g = green (verde)

b = blue (azul)

O valor CSS expresso pela função `rgb(red, green, blue)` indica uma cor obtida com a mistura de uma quantidade *red* da cor vermelha com *green* da cor verde e *blue* da cor azul.

Duas são as maneiras de se definir a quantidade de cada uma das três cores (os parâmetros da função):

Uma faixa de números de 0 (zero) até 255

Uma faixa de porcentagens de 0% até 100%

Não é válido usar em uma definição de cor uma mistura de números e porcentagens. Observe a seguir as duas sintaxes válidas e uma inválida.

CSS

```
/* Sintaxes válidas */
```

```
rgb(145, 230, 50)
```

```
rgb(20%, 0%, 70%)
```

```
/* Sintaxe inválida */
```

```
rgb(255, 20%, 120)
```

Declaração de cores com uso da função CSS `rgba()`

O Módulo CSS3 para Cores estendeu a função `rgb()` criando a função `rgba()` (red, green, blue, alpha-opacity) acrescentando mais um parâmetro na declaração da cor, destinado a definir a opacidade em uma faixa de valores decimais de 0 até 1. Os valores RGB podem ser declarados em uma faixa numérica de 0 até 255 ou percentual de 0 até 100%, conforme descrito na seção anterior.

Para que você possa constatar o funcionamento da declaração de valores de cores com uso da função `rgba()` desenvolvi uma tela interativa que é apresentada a seguir. O funcionamento é intuitivo, mas aqui vão as instruções gerais:

- A interação só vai funcionar (é claro) em navegadores que suportam RGBA, ou seja, os IE8 e anteriores estão fora;
- Os campos RGB (red, green, blue) admitem valores numéricos de 0 a 255. Não contemplam porcentagens embora sejam válidas;

[topo](#)

- O campo A (opacidade alfa) admite decimais de 0 a 1 sendo 0 transparente e 1 opaco. Valores decimais devem ser escritos com uso de ponto, sendo o 0(zero) inicial opcional. Exemplo: 0.5 ou .5
- Para valores RGB a especificação prevê que valores superiores a 255 serão considerados 255 e inferiores a zero (números negativos) serão considerados 0. Contudo não há razão para usar valores fora da faixa e o script interativo não admite tais valores;
- Para valores A (opacidade) a especificação prevê o mesmo comportamento do item anterior com relação aos limites 0 e 1;
- O retângulo na cor preta que se encontra na área interativa destina-se a facilitar a visualização da opacidade aplicada.
- Aprenda divertindo-se! 😊

RGBA Interativo

Altere os valores RGBA nos campos abaixo e clique OK.

R: G: B: A:

A cor RGBA escolhida será aplicada como fundo deste retângulo

com uso da função CSS `hsl()`

A declaração de cores com uso da função `hsl(hue, saturation, lightness)`, não existia nas CSS2.1, Ela permite que você declare as cores com uso de três parâmetros:

Hue = tom, Saturation = saturação e Lightness = luminosidade.

A sintaxe geral para a declaração com uso desta função é mostrada a seguir:

CSS

```
seletor { color: hsl(120, 75%, 50%); }
```

O primeiro valor é para o tom (hue) da cor. O seu valor é um número que representa a medida de um ângulo (expresso em graus) apontando para um tom da cor na roda de cores. Observe os valores a os respectivos tons de cores:

CSS

[topo](#)

```

seletor { color: hsl(0, 100%, 50%) } /* vermelho */
seletor { color: hsl(120, 100%, 50%) } /* verde */
seletor { color: hsl(120, 100%, 25%) } /* verde escuro */
seletor { color: hsl(120, 100%, 75%) } /* verde claro */
seletor { color: hsl(120, 75%, 75%) } /* verde pastel */
seletor { color: hsl(60, 100%, 50%) } /* amarelo */
seletor { color: hsl(240, 100%, 50%) } /* azul */
seletor { color: hsl(300, 100%, 50%) } /* púrpura */

```

hsl(0,100%,50%) - vermelho

hsl(60,100%,50%) - amarelo

hsl(120,100%,50%) - verde

hsl(180,100%,50%) - ciano

hsl(240,100%,50%) - azul

hsl(300,100%,50%) - púrpura

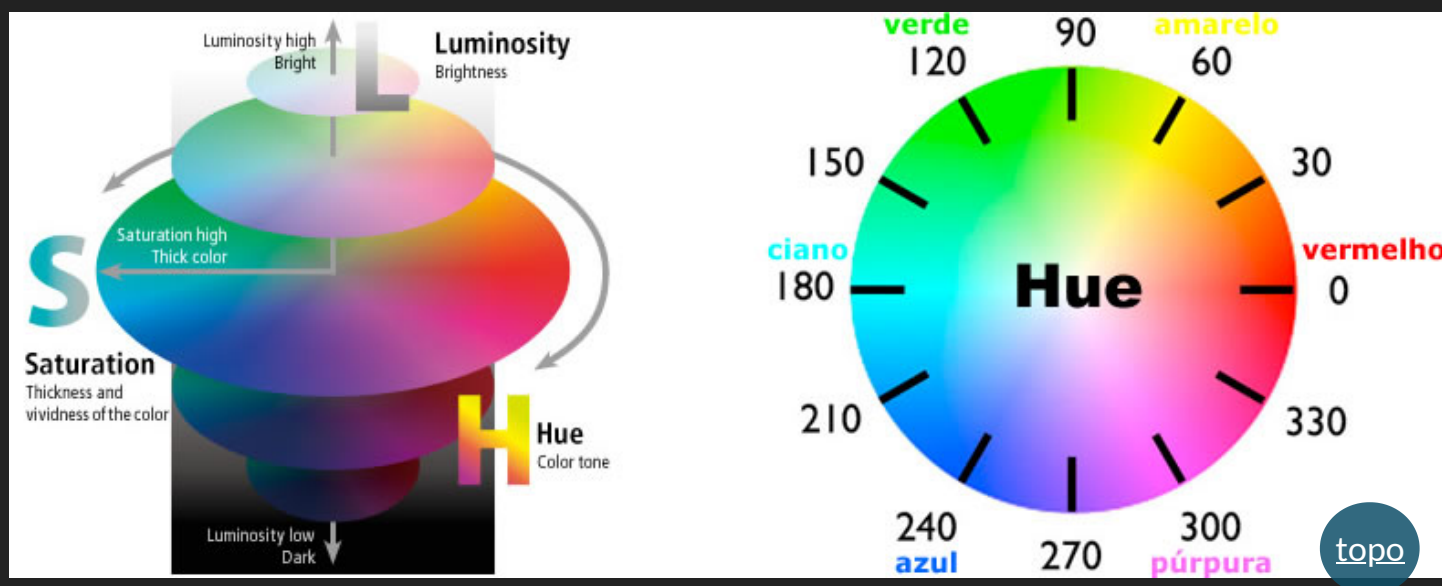
hsl(360,100%,50%) - vermelho

O valor do ângulo segue as regras da trigonometria para medida de ângulos. Por exemplo 0graus = 360graus, -120graus = 240graus e assim por diante.

O segundo valor é para a saturação (saturation) da cor. O seu valor é expresso em porcentagem. Um valor igual a 100% representa saturação total da cor e 0 é um leve sombreado cinza de saturação.

O terceiro valor é para a luminosidade (lightness). O seu valor é expresso em porcentagem. Um valor igual a 100% resulta em cor branca e 0 em cor preta, sendo 50% o valor normal.

As figuras a seguir dão uma idéia bem aproximada de como os valores determinam a cor final.



Como você já deve ter notado o uso da função `hsl()` para declarar cores é semelhante ao uso da função `rgb()`. A diferença é que HSL é uma maneira mais intuitiva de se declarar cores.

Declaração com uso da função `hsla()`

A declaração de cores com uso da função `hsla(hue, saturation, ightness, alpha-opacity)` é uma maneira estendida da declaração com uso da função `hsl` na qual um quarto argumento define a opacidade da cor. Este quarto argumento é um número decimal na faixa de 0 até 1.

A seguir a versão interativa para você fazer experiências com a declaração de cores usando a função `hsla()`.

- A interação só vai funcionar (é claro) em navegadores que suportam HSLA, ou seja, os IE8 e anteriores estão fora;
- O campo H (hue) admite valores de 0 a 360. Não contempla valores maiores que 360 e nem valores negativos embora sejam válidos;
- O campo A (opacidade alfa) admite decimais de 0 a 1 sendo 0 transparente e 1 opaco. Valores decimais devem ser escritos com uso de ponto, sendo o 0(zero) inicial opcional. Exemplo: 0.5 ou .5
- Para valores A (opacidade) a especificação prevê o mesmo comportamento do item anterior com relação aos limites 0 e 1;
- O retângulo na cor preta que se encontra na área interativa destina-se a facilitar a visualização da opacidade aplicada.
- Aprenda divertindo-se! 😊

HSLA Interativo

Altere os valores HSLA nos campos abaixo e clique OK.

H: S: % L: % A:

A cor HSLA escolhida será aplicada como fundo deste retângulo

Gradientes

Uma aplicação prática bem interessante para uso das funções `rgba()` ou `hsla()` é a possibilidade de se criar uma máscara transparente a ser aplicada sobre uma cor sólida obtendo como resultado o efeito gradiente.

[topo](#)

Você cria um componente padrão para sua aplicação estilizado com diferentes cores de fundo em gradiente de acordo com o conteúdo ou finalidade do componente como mostrado no exemplo a seguir.

HTML

```
<div class="box perfumes">
  PERFUMES
</div>
<div class="box eletronicos">
  ELETRÔNICOS
</div>
<div class="box moveis">
  MÓVEIS
</div>
```

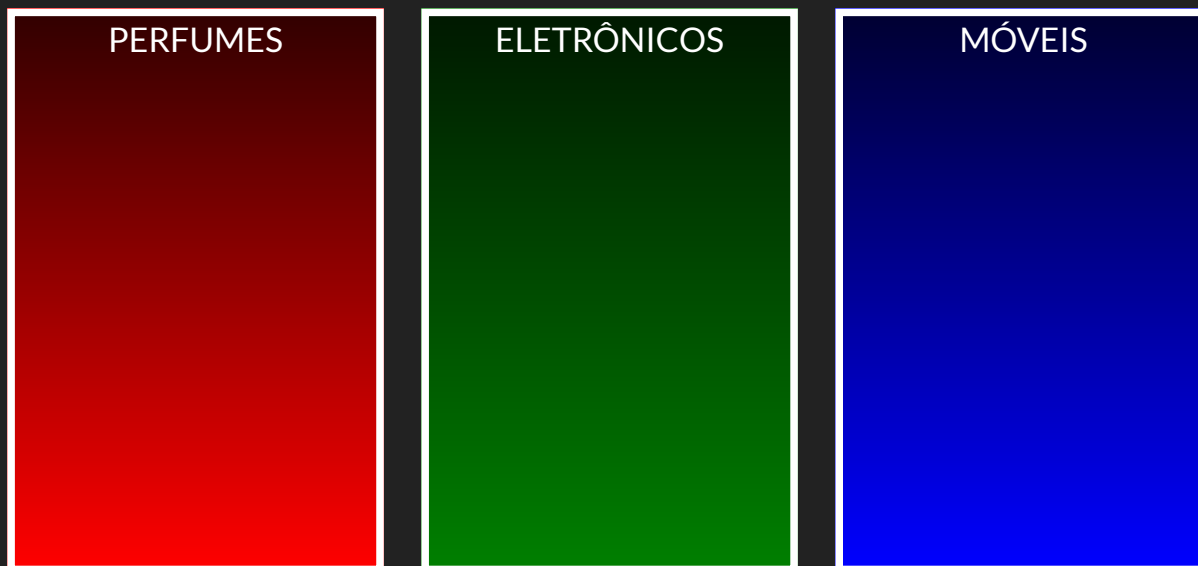
CSS

```
div.box {
  float:left;
  text-align: center;
  margin-right:15px;
  width: 200px;
  height: 300px;

  border: 4px solid white;
  background: linear-gradient(to bottom, rgba(0, 0, 0, 0.8), transparent);
}
div.box.perfumes {
  background-color: red;
}
div.box.eletronicos {
  background-color: green;
}
div.box.moveis {
  background-color: blue;
}
```

[topo](#)

O que resulta na estilização dos boxes "para perfumes", "para eletrônicos", "para móveis" respectivamente, mostrados a seguir:



Nesse exemplo, para demonstrar a técnica da máscara usamos a função `rgba()`, mas o uso da função `rgb()` produz o mesmo efeito. É uma questão de preferência pessoal.

Sugestão: Crie máscaras com gradientes de mais de duas cores e também com gradientes radiais.

Conclusões

O uso das declarações CSS para cores com estas novas funcionalidades e em particular a possibilidade de declarar opacidade alfa simplifica, e muito, declarar cores com CSS. Notar que os conteúdos inseridos em um container com opacidade assim declarada não herdam a opacidade, ou seja, os textos dentro de containers transparentes, bem como suas bordas não ficam transparentes. Além disso usar a técnica de criação de máscara para simular gradientes é uma ferramenta poderosa para simplificação de código, modularização das CSS e criação de componentes.

Conheça os livros do Maujor®

Ir para a [página de entrada nos sites dos livros](#).

Links patrocinados

Blog do Maujor

Editor de fotos online e gratuito

 Adobe Spark

Fundamentos CSS

[topo](#)