Web Design

CSS: Cascade Style Sheet

03 - Cores e Backgrounds

Prof. Me. Lucas Mendes | GitHub: @lucas-lfm

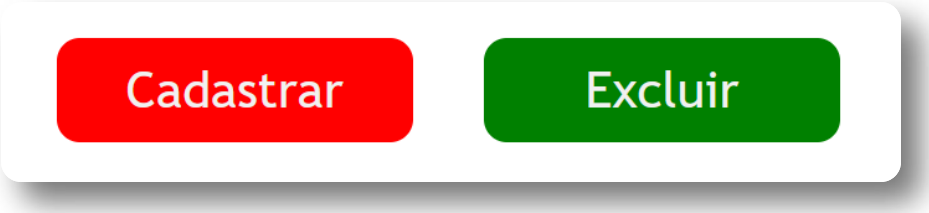
Roteiro

- Cores
- Backgrounds

Cores

> As cores transmitem sentimentos, então escolha bem

- Antes de entrar na parte técnica de CSS, que tal entender um pouquinho de **teoria das cores**?
- Veja o exemplo abaixo, onde temos dois botões com cores diferentes e representando ações distintas. Qual sua opinião sobre a escolha das cores aplicadas?



Cores

> Um **pouquinho** de teoria das cores

- Teoria das cores é o estudo sobre vários **aspectos das cores**, inclusive como elas **são interpretadas pelo nosso cérebro**
- Ter pelo menos um conhecimento mínimo sobre teoria das cores é fundamental para sua correta utilização em meios digitais como a Web
- Uma escolha adequada do conjunto/combinação de cores a ser utilizada em um projeto visual **não é uma tarefa trivial**, mas ajuda a **alcançar o objetivo** do projeto e entregar uma **boa experiência ao usuário**

Cores

> Um **pouquinho** de teoria das cores

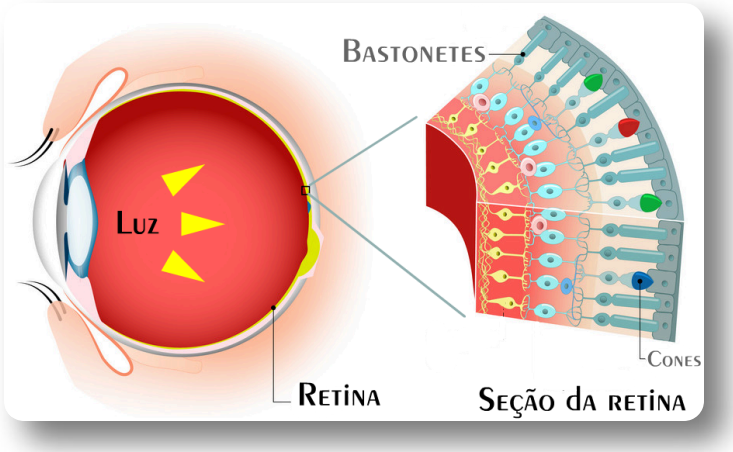
- A escolha correta das cores, depende da **mensagem** que se quer transmitir e também do **público alvo**
- As cores **transmitem emoções** para as pessoas, na maioria das vezes de forma **inconsciente**
- Acesse aqui um artigo sobre Psicologia das Cores e os significados comumente associados às cores mais comuns

> 💡 **Importante saber**
>
> Uma cor pode despertar um sentimento em uma pessoa e pode despertar o sentimento oposto em outra pessoa. Isso pode ocorrer devido à diferenças culturais, experência prévia ou até mesmo por divergências de gostos pessoais.

Cores

> O que são cores?

- Podemos dizer que cor **não existe de forma tangível**
- Cor é uma sensação produzida pelo olho a partir da luz refletida pelos objetos
- Segundo a Teoria Tricromática de Young-Helmholtz, a retina possui três espécies de células sensíveis à comprimentos de onda específicos, correspondendo às cores vermelho, verde e azul



Cores

> Classificação das cores

- Dessa forma, dentro da teoria das cores temos uma classificação bastante utilizada para composição de cores, onde dividem-se em:
  - **Cores primárias:** cores que não podem ser decomposta e são a base para a formação das outras cores
  - **Cores secundárias:** cores formadas a partir da composição das cores primárias
  - **Cores terciárias:** cores formadas a partir da composição de cores secundárias e primárias

Cores

Classificação das cores

- Dessa forma, dentro da teoria das cores temos uma classificação bastante utilizada para composição de cores, onde dividem-se em:
  - **Cores primárias:** cores que não podem ser decor
  - **Cores secundárias:** cores formadas a partir da co
  - **Cores terciárias:** cores formadas a partir da comp

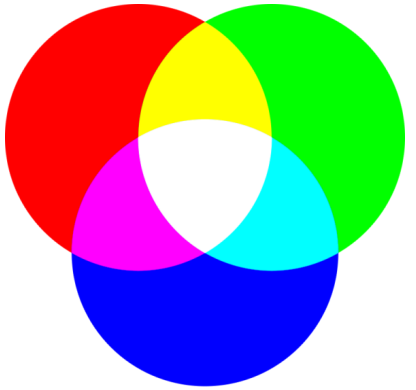
Existem diferentes sistemas de cores primárias. Na disciplina vamos abordar principalmente o sistema **RGB** (abordado mais à frente).

Para saber mais sobre esse assunto, acesse esse link.

Cores

Sistema RGB

- Sistema de cores **aditivas** (a luz é emitida pela tela)
- Tem como base o triângulo de cores de James C Maxwell
- Formado pelas três cores primárias: (**R**)ed, (**G**)reen e (**B**)lue

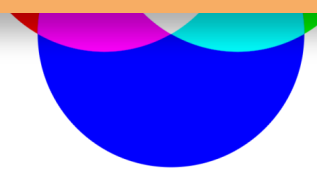


Cores

Sistema RGB

- Sistema de cores **aditivas** (a luz é emitida pela tela)
- Tem como base o triângulo de cores de James C Maxwell
- Formado pelas três cores primárias: (**R**)ed, (**G**)re

> Na perspectiva da cor como luz, nesse sistema, a mistura de duas cores resultará sempre em uma cor mais luminosa. Logo, ao misturar as três cores primárias na intensidade máxima, o resultado é o branco.



Cores

Função RGB

- Para definir uma cor no CSS, com o sistema RGB, utilizamos a função ***rgb()***
- 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.
- Temos duas maneiras de definir a quantidade de cada cor na função ***rgb()***:
  - Uma faixa de números de 0 (zero) a 255
  - Uma faixa de porcentagens de 0% a 100%

```
/* Sintaxe - rgb()*/
color: rgb(185, 245, 35);
color: rgb(30%, 85%, 15%);
```

Cores

Função RGBA

- O CSS3 estendeu a função ***rgb()*** criando a função ***rgba()*** *(red, green, blue, alpha-opacity)*, acresentando mais um parâmetro na declaração da cor, destinado a definir a **opacidade** em uma faixa de valores decimais **de 0 até 1**.

```
/* Sintaxe - rgba()*/
color: rgba(185, 245, 35, 0.5); /* 50% de opacidade */
color: rgba(30%, 85%, 15%, 0.8); /* 80% de opacidade */
```

Cores

> Padrão Hexadecimal

- O padrão hexadecimal especifica uma cor com: **#RRGGBB**
  - **RR** (red), **GG** (green) e **BB** (blue) são **inteiros hexadecimais** que especificam a quantidade de cada cor no sistema RGB
  - **valores entre 00 e FF** para cada cor (da mesma forma que no decimal o intervalo é 0-255)
  - por exemplo, o código **#FF0000** define a cor vermelha, pois o componente da cor vermelha (RR) foi definido com valor máximo (FF) e os demais componentes com o menor valor (00)

```
/* Sintaxe - padrão hexa*/
color: #F6FCFF; /* RR -> F6, GG -> FC, BB -> FF */
```

Cores

> Padrão Hexadecimal

- No padrão hexadecimal, também é possível especificar um valor para a opacidade da cor
- Para isso, adicionamos dois dígitos hexadecimais ao final do código da cor para especificar o grau de opacidade/transparência dessa cor

```
/* Sintaxe - #RRGGBBAA */
color: #FF4264AA; /* os dígitos AA, ao final, definem o grau de opacidade */
```

Cores

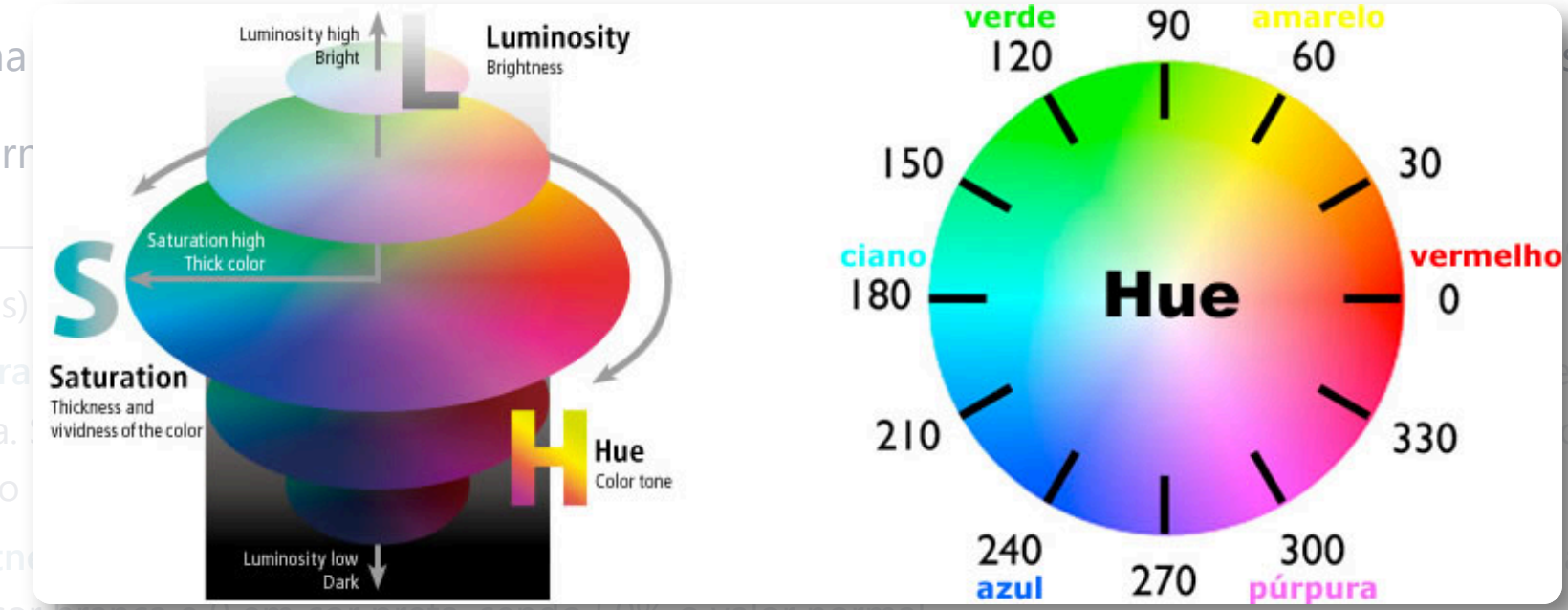
> Sistema HSL

- O sistema **HSL** é representado por **H**ue (tom), **S**aturation (saturação) e **L**ightness (luminosidade)
- Dessa forma, podemos declarar as cores com uso desses três parâmetros:
  - ***hue*** — especifica o **tom 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**.
  - ***saturation*** — especifica a **saturação da cor**. O seu valor é expresso em porcentagem. Quanto menos cor, mais cinza. Se você quiser uma cor mais suja, mais apagada, você diminui este valor. Caso contrário você a mantém como 100% e utiliza a quantidade integral da cor.
  - ***lightness*** — especifica a **luminosidade**. 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.

Cores

Sistema HSL

- O sistema
- Dessa forr
  - hue —
  - satura
  - lightn



Cores

Função HSL e função HSLA

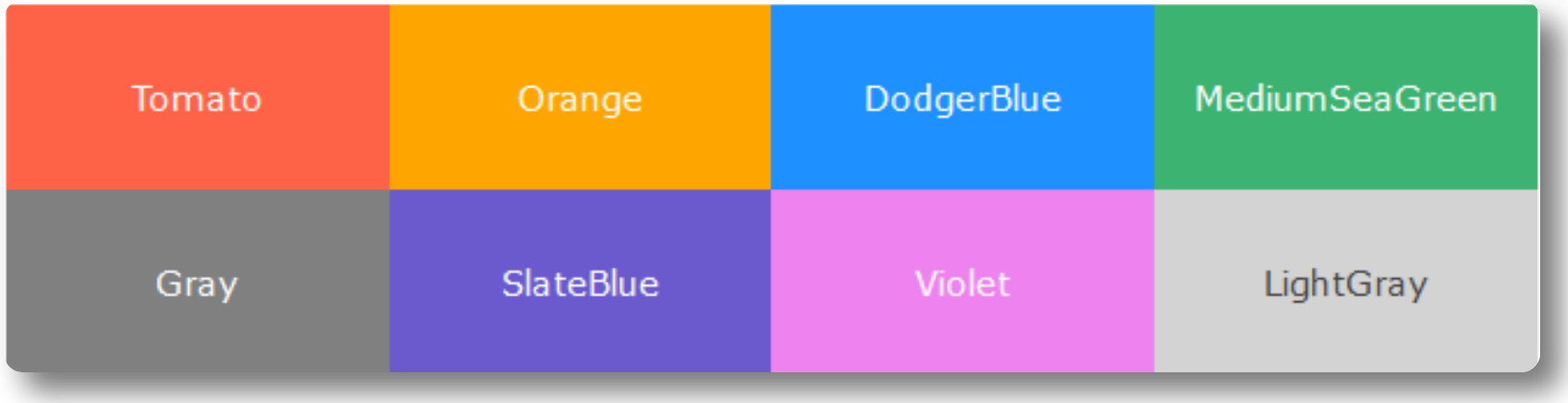
- Para declarar um cor no CSS usando o padrão HSL, usamos a função ***hsl()***, passando os três parâmetros necessários
- Também podemos definir o grau de opacidade da cor com a função ***hsla()*** adicionando um parâmetro para o canal alfa, com valor variando de 0 a 1

```
/* Sintaxe - hsl(hue, saturatiom, lightness)*/
color: hsl(120, 100%, 50%); /* hue (em graus), saturation (porcentagem) e lightness (porcentagem) */
color: hsla(120, 100%, 50%, 0.5) /* 50% de opacidade */
```

Cores

Nome das cores

- Também podemos usar um nome predefinido para declarar uma cor



- O CSS suporta 140 nomes de cores predefinidas

Cores

> Principais Propriedades

- Entre as principais propriedades CSS onde podemos trabalhar com cores, estão:
  - `background-color` — define uma cor de plano de fundo para um elemento HTML
  - `color` — define a cor do texto de um elemento
  - `border-color` — define a cor da borda de um elemento
  - `text-shadow` e `box-shadow` — define efeito de sombra em textos ou em elementos, onde podemos especificar a cor da sombra

Cores

Gradientes

- Gradientes CSS permitem definir transições suaves entre dois ou mais cores especificadas
- Podemos trabalhar com três tipos de gradientes:
  - **Gradientes Lineares** — a transição ocorre segundo um eixo (linha reta) que possui um sentido e uma direção
  - **Gradientes Radiais** — a transição parte de um ponto central e se espalha em direção às bordas
  - **Gradientes Cônicos** — a transição é circular usando o centro do elemento como ponto de origem

Cores

Gradientes Lineares

- Para criar gradientes lineares devemos definir pelo menos duas cores
- Também podemos definir um ponto inicial e uma direção (ou um ângulo)
- **Sintaxe Básica:**

```
background-image: linear-gradient(direcao, color-stop1, color-stop2, ...);
```

- Note que usamos a propriedade `background-image`
  - Isso porque um gradiente CSS nada mais é do que uma imagem cirada pela função gradiente do CSS

Cores

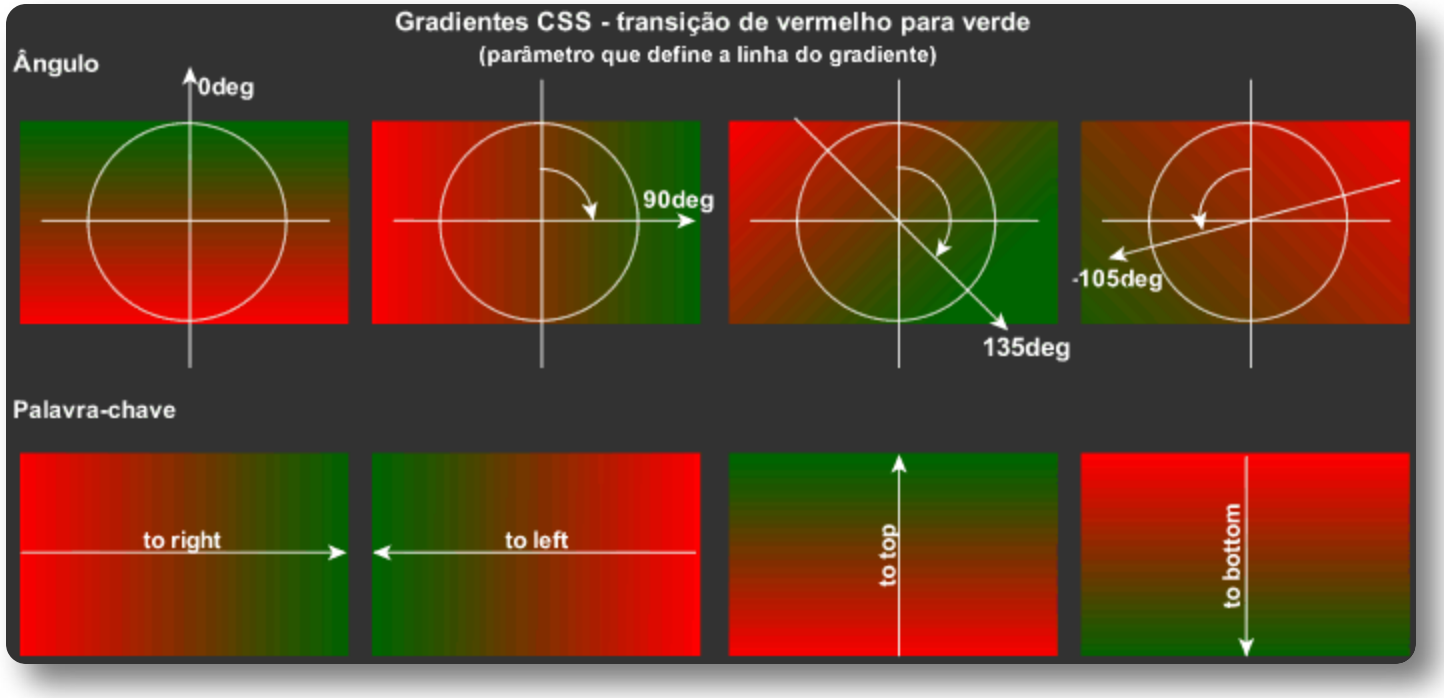
Gradientes Lineares

- A **linha do gradiente** é definida pelo primeiro parâmetro (direção e sentido), podendo ter como valor um ângulo (expresso em *deg*, por exemplo) ou uma das seguintes *palavras-chave*:
  - **to right** → transição de cores na horizontal da esquerda para a direita;
  - **to left** → transição de cores na horizontal da direita para a esquerda;
  - **to top** → transição de cores na vertical de baixo para cima;
  - **to bottom** → transição de cores na vertical de cima para baixo;
  - **to top left** → transição de cores a partir do canto superior esquerdo em direção ao centro;
  - **to top right** → transição de cores a partir do canto superior direito em direção ao centro;
  - **to bottom left** → transição de cores a partir do canto inferior esquerdo em direção ao centro;
  - **to bottom right** → transição de cores a partir do canto inferior direito em direção ao centro.

Cores

Gradientes Lineares

- A figura abaixo traz uma relação entre transições segundo um ângulo e uma palavra-chave



Cores

Gradientes Lineares

- O segundo parâmetro define as cores do gradiente e, *opcionalmente*, a posição onde a cor começa no gradiente, como a seguir:

```
background-image: linear-gradient(90deg, rgba(2,0,36,1) 0%, rgba(9,9,121,1) 35%, rgba(0,212,255,1) 100%);
```

Cores

> Gradientes Radiais

- Para criar gradientes radiais é necessário definir o ponto central, o formato e as cores que serão utilizadas
- **Sintaxe Básica:**

```
background-image: radial-gradient(shape size at position, start-color, ..., last-color);
```

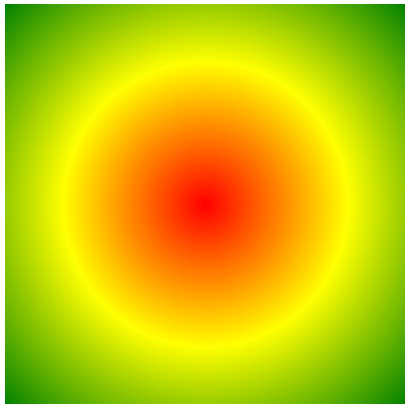
- Por padrão, o formato (*shape*) é elíptico (*ellipse*) , o tamanho (*size*) é ***farthest-corner*** e a posição é ***center***

Cores

Gradientes Radiais

- O parâmetro ***shape*** define o formato do gradiente, que pode ser **circle** ou **ellipse**
- Exemplo:

```
#grad {
    background-image: radial-gradient(circle, red, yellow, green);
}
```



Cores

Gradientes Radiais

- O parâmetro ***size*** define o tamanho do gradiente, que pode ter quatro valores:
  - **closest-side**
  - **farthest-side**
  - **closest-corner**
  - **farthest-corner**
- Exemplos: W3Schools

Cores

Gradientes Radiais

- Experimente diferentes formatos, direções, transparências e múltiplos gradientes para criar degradês mais complexos, como no exemplo abaixo que está aplicado a este slide.

```
body {
  background: radial-gradient(ellipse at top, #e66465, transparent),
              radial-gradient(ellipse at bottom, #a020f0, transparent),
              radial-gradient(circle at 10%, #1974d1, transparent);
}
```

Cores

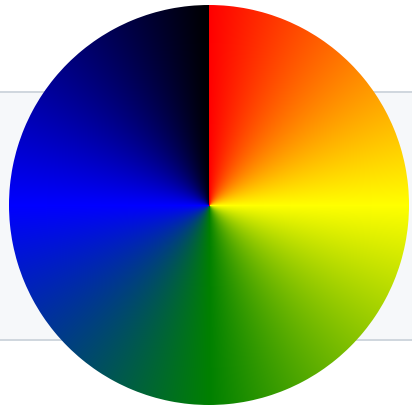
Gradientes Cônicos

- Para criar um gradiente cônico, devemos definir pelo menos duas cores, seguindo a sintaxe básica:

```
background-image: conic-gradient([from angle] [at position,] color [degree], color [degree], ...);
```

- Por padrão o ângulo é 0 (zero) e a posição é **center**
- Exemplo (gráfico de pizza):

```
#grad {
    background-image: conic-gradient(red, yellow, green, blue, black);
    border-radius: 50%;
}
```



Cores

Ferramentas Úteis

- **Ferramentas e sites úteis** para explorar cores, gradientes e verificações de acessibilidade:
  - Gerar paleta e combinação de cores:
    - *https://coolors.co/*
    - *https://color.adobe.com/pt/create*
  - Gradientes CSS:
    - *https://cssgradient.io/*
    - *https://uigradients.com/*
    - *https://uigradients.com/*
  - Acessibilidade:
    - *https://helpx.adobe.com/br/creative-cloud/adobe-color-accessibility-tools.html*
    - *https://color.adobe.com/pt/create/color-contrast-analyzer*

Backgrounds

> Propriedades Background

- As propriedades de **background CSS** são usadas para adicionar efeitos ao plano de fundo dos elementos
- Destacam-se as seguintes propriedades:
  - `background-color`
  - `background-image`
  - `background-repeat`
  - `background-size`
  - `background-attachment`
  - `background-position`
  - `background` (*shorthand property*)

Backgrounds

> Propriedade `background-image`

- A propriedade **`background-image`** especifica uma imagem para usar como plano de fundo de um elemento
- Também podemos usar essa propriedade para criar gradientes em elementos, como visto anteriormente
- Sintaxe básica:

```
background-image: url("image.jpg");
background-image: url("image1.jpg", "image2.jpg"); /* adicionando duas imagens ao background */
```

Backgrounds

> Propriedade `background-repeat`

- Por padrão, as imagens são repetidas para cobrir todo o background do elemento (tanto na horizontal como na vertical)
- Podemos alterar esse comportamento através da propriedade `background-repeat`
- Sintaxe básica:

```css
background-image: url("image.jpg");
background-repeat: repeat; /* padrão */
background-repeat: repeat-x; /* somente no eixo horizontal */
background-repeat: repeat-y; /* somente no eixo vertical */
background-repeat: no-repeat; /* sem repetição */
```

Backgrounds

Propriedade `background-position`

- A propriedade **`background-position`** define a posição da imagem no plano de fundo
- Podemos usar as palavras-chave `top`, `bottom`, `left`, `right` e `center`
- Podemos ainda usar valores em porcentagem ou alguma unidade de medida para especificar a posição
- No caso de múltiplas imagens, podemos definir a posição de cada uma, separando por vírgula
- Sintaxe básica:

```css
background-position: center;
background-position: bottom 10px right 20px;
background-position: 1cm 2cm;
background-position: 0 0, center; /* múltiplas imagens */
```

Backgrounds

Propriedade `background-size`

- A propriedade **`background-size`** define o tamanho da imagem de plano de fundo
- Espaços não cobertos pela imagem são preenchidos com o `background-color`
- Sintaxe básica e valores:

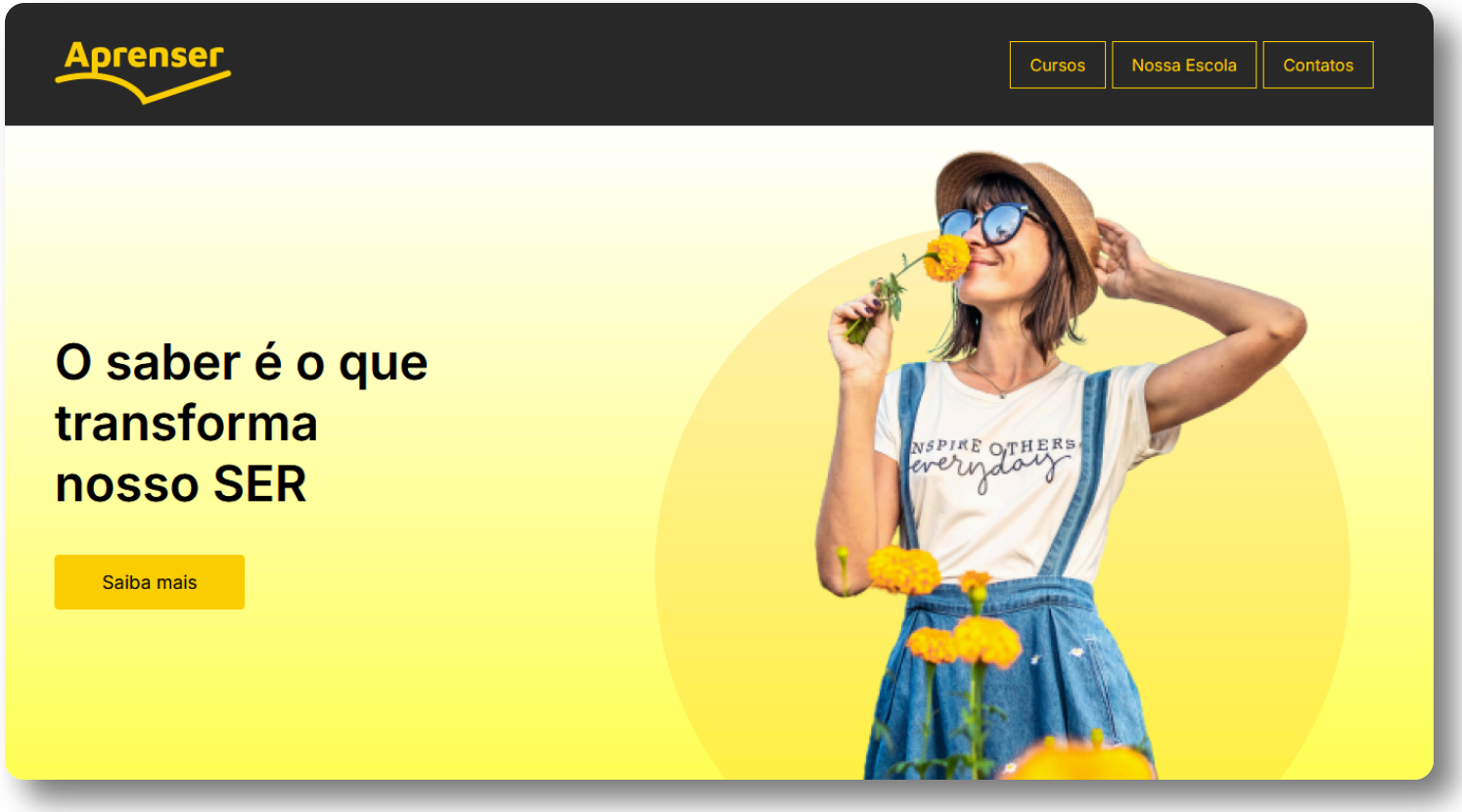
```css
background-size: contain; /* redimensiona a imagem para cobrir todo o contêiner */
background-size: cover; /* redimensiona a imagem para que fique totalmente visível */
background-size: 50%; /* define a largura da imagem */
background-size: 50% 3em; /* define largura e altura da imagem */
```

- Veja mais em W3Schools

Mãos à Obra 

Atividade Prática

Vamos replicar o resultado abaixo aplicando o que aprendemos até aqui.



Isso é tudo, pessoal!

Sigam-me nas redes abaixo 👇😉

@lucas-lfm

@prof_lucasmendes

@prof-lucasmendes

Email: lucas.mendes@ifce.edu.br
Prof. Me. Lucas Mendes | GitHub: @lucas-lfm
Web Design 👷

Referências

- https://www.w3schools.com
- https://developer.mozilla.org/pt-BR
- https://desenvolvimentoparaweb.com
- https://github.com/gustavoguanabara/html-css
- https://maujor.com

Email: lucas.mendes@ifce.edu.br
Prof. Me. Lucas Mendes | GitHub: @lucas-lfm