

Módulo IV - Aprendendo CSS

Introdução ao CSS

O professor

Adriano Rivolli

- Desenvolvedor web desde 2002
- Professor (UTFPR)
- Mestrado e doutorado (USP)

O que é CSS?

- *Cascading Style Sheets*
- Folha de Estilo em Cascatas
- Descreve como os elementos HTML devem ser apresentados
- Define a apresentação (*layout* e estilo) de uma página

O que é CSS?

Cascading Style Sheets

Folha de Estilo em Cascatas

Descreve como os elementos
HTML devem ser apresentados

Define a apresentação (*layout* e
estilo) de uma página

Exemplo

https://eastmanjian.github.io/HTML_CSS_Demo/css_basic_demo.html

Vantagens

- Separação entre conteúdo e formato
- Fácil e rápido de alterar
- Reutilização
- Responsividade

Ferramentas

- Navegador web
- VS Code
- Codepen (<https://codepen.io/pen/>)

Material de apoio

- W3 Schools (<https://www.w3schools.com/css>)
- Mozilla Developer (<https://developer.mozilla.org/pt-BR/docs/Web/CSS>)
- Google :)

Como funciona o CSS?

- **Seletor:** Define quais os elementos serão afetados
- **Regra:** Define como os elementos serão afetados
 - **Propriedade:** Define a característica que será modificada
 - **Valor:** Define como a característica será modificada

```
selector {  
  propriedade1: valor1;  
  propriedade2: valor2;  
  ...  
}
```

```
a {  
  color: blue;  
  font-size: 15px;  
}
```

Sites interessantes...

- Desafio dos 100 dias (<https://100dayscss.com/>)
- Batalha de CSS (<https://cssbattle.dev/>)

Próximas aulas

- Ferramentas
- Seletores
- Sintaxe
- Cores
- Medidas

Módulo IV - Aprendendo CSS

Ferramentas

Navegador

- Chrome
- Edge
- Firefox
- Safari
- Opera
- ...

Chrome - Inspeção de código fonte

- Permite visualizar (estudar) as regras, propriedades e valores aplicados aos elementos
- Pressione F12
- Botão invertido (inspecionar elemento)

Visual Studio Code

- Editor de códigos HTML/CSS
- Site oficial:
 - <https://code.visualstudio.com/>
- Permite a instalação de *plugins*

Codepen

- Editor de códigos HTML/CSS online
- Site:
 - <https://codepen.io/pen/>
- Permite compartilhar o código:
 - <https://codepen.io/rivolli/pen/yLvXdMR>

Módulo IV - Aprendendo CSS

Aplicando CSS

Introdução

- O que é a sintaxe de uma linguagem?
- Seletores e regras (propriedade e valor)
- Diretamente no elemento
- Internamente ao arquivo
- Arquivo externo

Sintaxe

```
selector {  
  propriedade1: valor1;  Regra 1  
  propriedade2: valor2;  Regra 2  
  ...  
}
```

Seletor de elemento

- É o seletor mais simples e genérico
- Basta definir o nome do elemento
- Todos os elementos com o respectivo nome será modificado
- Exemplo:

```
a {  
  ...  
}
```

```
div {  
  ...  
}
```

```
h1 {  
  ...  
}
```

Regras

- Cor do texto
 - `color: red;`
- Sublinhado
 - `text-decoration: underline;`
- Cor do fundo
 - `background-color: green;`
- Borda
 - `border: 1px solid black;`

Diretamente no elemento

- Aplicado a um único elemento e seus descendentes
- Mais específico entre as alternativas
- Têm a maior prioridade
- Dispensa o seletor
- Atributo **style**
- Exemplo:
 - `<h1 style="color: red">Texto em vermelho</h1>`
 - `<h1 style="color: red; text-decoration: underline;">Texto em vermelho e sublinhado</h1>`

Internamente no arquivo

- Aplicado a uma única página
- Elemento **<style>** normalmente definido no **<head>** da página
- Exemplo:

```
<style type="text/css">
```

```
  p { color: blue; }
```

```
</style>
```

Arquivo externo

- Pode ser incluído em qualquer página
- Elemento **<link>** normalmente definido no **<head>** da página
- Exemplo:

```
<link rel="stylesheet" type="text/css" href="arquivo.css" />
```


Módulo IV - Aprendendo CSS

Seletores: ID e Classe

Seletores

- São utilizados para selecionar os elementos
- Seletor de elemento se aplica para a um tipo específico de elemento
- Identificadores são utilizados para distinguir um elemento
- Classes são utilizados para distinguir um grupo de elementos

ID

- Todo elemento deve possuir um valor único na página
- São únicos em uma página HTML
 - `<div id="conteudo"></div>`
- Os seletores de ID são precedidos pelo símbolo #
 - `#conteudo { ... }`
 - `#meuid { ... }`

Classes

- Agrupa os elementos de acordo com alguma característica
- A mesma classe pode ser aplicada em diferentes elementos HTML
 - `<div class="importante"></div>`
- Os seletores de classe são precedidos pelo símbolo `.`
 - `.importante { ... }`
 - `.comentario { ... }`
 - `.alerta { ... }`

Módulo IV - Aprendendo CSS

Combinando Seletores

Template

<https://codepen.io/rivolli/pen/xxYLbbE>

Seletor universal

- Utilizado para selecionar todos os nós
- As regras serão aplicados a todos os elementos
 - `* { ... }`

Escrevendo regras em partes

- É possível definir a mesma regra para diferentes seletores
- Basta separar os seletores por vírgula
 - `h1, #principal, .importante { ... }`
- Geralmente os seletores são depois especializados

Especializando os seletores

- É possível combinar os seletores de elemento com os seletores de ID e classe
- Faz com que o seletor se torne mais específico
- Basta utilizar o nome do elemento antes do símbolo # ou .
 - `h1#principal { ... }`
 - `p.importante { ... }`

Seletor de descendente e de filho

- O combinador seleciona os nós que são descendentes ou filhos diretos de outros elementos
- Para descendência basta utilizar um espaço entre os seletores
 - `#principal .importante { ... }`
 - `p span { ... }`
- Para filho direto basta utilizar um símbolo de maior `>` entre os seletores
 - `#principal > .importante { ... }`
 - `p > span { ... }`

Módulo IV - Aprendendo CSS

Seletores de atributos

Template

<https://codepen.io/rivolli/pen/ExQvavZ>

Comentários

- Permite realizar anotações no código sem que elas sejam interpretadas
- Os comentários são realizados dentro de blocos
 - `/*` Qualquer texto escrito aqui será ignorado `*/`

Seletor de atributo

- Permite realizar a seleção a partir do valor ou da presença dos atributos
- Apresenta as seguintes variações:
 - Possui um atributo
 - Possui exatamente um determinado valor para o atributo
 - Contém uma palavra específica no atributo
 - Começa com uma sequência específica
 - Termina com uma sequência específica

Exemplos

```
/* Elementos <a> que possuem um atributo title */
```

```
a[title] {  
  color: purple;  
}
```

```
/* <a> que possuem o valor de href igual a "https://example.org" */
```

```
a[href="https://example.org"] {  
  color: green;  
}
```

```
/* Elementos <a> que href contém "example" */
```

```
a[href*="example"] {  
  font-size: 2em;  
}
```



Combinando seletores

- Os seletores de atributos também podem ser combinados com os demais elementos

Módulo IV - Aprendendo CSS

Pseudo-classes

Template

<https://codepen.io/rivolli/pen/ExQvavZ>

Pseudo-classes

- Utilizado para definir um estado específico de um elemento
- Exemplos:
 - Quando o cursor do mouse está sobre o elemento
 - Quando um link já foi visitado
 - Quando um elemento recebe o foco da ação

Sintaxe

```
seletor:pseudo-class {
```

```
...
```

```
}
```

Pseudo classes para os links

- Link não visitado
 - `a:link { ... }`
- Link visitador
 - `a:visited { ... }`
- Mouse sobre o link (também funciona com outros elementos)
 - `a:hover { ... }`
- Link selecionado
 - `a:active`

Elementos de formulário

- Quando um elemento está com foco
 - `input:focus`
- Quando um elemento é selecionado
 - `input:checked`
- Quando um elemento é obrigatório
 - `input:required`

Outras pseudo classes

- Primeiro filho de um elemento
 - :first-child
- Último filho de um elemento
 - :last-child
- Único filho
 - :only-child
- Quando um elemento está desabilitado
 - :disabled
- Elementos sem filhos
 - :empty

Módulo IV - Aprendendo CSS

Pseudo-elementos

Pseudo-elementos

- Utilizado para formatar partes de um elemento
- Exemplos:
 - Primeira letra
 - Primeira linha do texto
 - O marcador de uma lista
 - A seleção do usuário

Sintaxe

```
seletor::pseudo-elemento {
```

```
...
```

```
}
```

Primeira letra

- Permite alterar a aparência da primeira letra
 - `p::first-letter { ... }`

Primeira linha

- Permite alterar a aparência da primeira letra
 - `div:first-line { ... }`

Alterar o marcador

- Permite alterar o marcador de uma lista
 - `ul::marker { ... }`

Seleção do usuário

- Permite alterar o texto selecionado pelo usuário
 - `div::selection { ... }`

Módulo IV - Aprendendo CSS

Especificidade

Seletores

https://www.w3schools.com/cssref/css_selectors.asp

Especificidade

- Um elemento pode possuir regras distintas e contraditórias
- O navegador considera o seletor mais específico
- Para isso ele usa uma pontuação de especificidade

Exemplos...

Hierarquia de especificidade

- Estilo de linha
 - `<h1 style="color: pink;">`
- IDs
 - `#navbar`
- Classes, pseudo-classes e atributos
 - `.test, :hover, [href]`
- Elementos e pseudo-elementos
 - `h1`

Como calcular a especificidade?

- A partir do valor 0
 - **Estilo de linha:** 1000
 - **IDs:** 100
 - **Classe/pseudo-classe/atributo:** 10
 - **Elemento:** 1
- O valor mais alto será aplicado
- Se o valor for igual a última regra definida será aplicada

Exemplos

Seletor	Especificidade	Cálculo
p	1	1
p.test	11	1 + 10
p#demo	101	1 + 100
<p style="color: pink;">	1000	1000
#demo	100	100
.test	10	10
p.test1.test2	21	1 + 10 + 10
#navbar p#demo	201	100 + 1 + 100
*	0	0 (ignorado)

Fonte: https://www.w3schools.com/css/css_specificity.asp

Burlando as regras

- É possível dizer ao navegador para não considerar a especificidade
- Utilize **!importante** nestes casos
 - `p { color: blue !important; }`
- Use apenas em casos muito específicos
- Evite sempre que possível, reescrevendo as regras e seletores

Módulo IV - Aprendendo CSS

Herança das características

Introdução

- Alguns elementos HTML possuem características visuais por padrão
- Os elementos também possuem algumas propriedades CSS definidas
- Isso define o que acontece com um elemento quando não há regras para definir sua formatação
- Existe um comportamento padrão dos elementos

Valores padrão

https://www.w3schools.com/cssref/css_default_values.asp

Exemplos...

Valor computado

- Algumas propriedades, se não especificadas, são herdadas para os descendentes
 - cor do texto
 - fonte
 - fundo
- Outras não são herdadas, pois são “resetadas” a cada nível
 - borda
 - margem

Valor inherit

- As propriedades que não são herdadas
- Podem herdar o valor do elemento ancestral por meio do valor inherit
- Exemplo:
 - `p.teste span { border: inherit; }`

CSS Reset

- Cada navegador tem sua apresentação por padrão
- Isso pode gerar alguns conflitos de apresentação
- Uma alternativa é sobrescrever todos os valores padrão removendo-os
- <https://necolas.github.io/normalize.css/>

Módulo IV - Aprendendo CSS

Box Model

Modelo de caixa (CSS Box Model)

- O conceito de Box Model é usado quando falamos sobre *layout*
- Todos os elementos HTML podem ser considerados como caixas
- As caixas possuem algumas características
 - Margem
 - Borda
 - Espaçamento interno
 - Conteúdo
- esse modelo permite definir o espaço entre os elementos e suas marcações

A caixa



Fonte: https://www.w3schools.com/css/css_boxmodel.asp

Conteúdo

- O conteúdo da caixa
- Apresentação de textos e imagens
- O conteúdo pode ser alterado por meio de regras CSS

Espaçamento interno

- Chamado de **padding**
- Uma área entre o conteúdo e a borda da caixa
- O *padding* é sempre transparente

Borda

- A borda define os limites da caixa
- Externo ao espaçamento interno e conteúdo
- A borda possui as seguintes propriedades
 - Tamanho
 - Estilo
 - Cor

Margem

- Espaçamento externo
- Define uma parte da distância entre a borda de 2 caixas
- A margem é sempre transparente

Tipos de caixa

- Blocos
- Em linha

Elementos de bloco

- A caixa gera uma nova linha, pois ocupa todo o espaço na horizontal
- Respeita as propriedades de altura e largura
- Elementos HTML de bloco:
 - ARTICLE, BLOCKQUOTE, DIV, DETAILS, FIELDSET, FOOTER, FORM, H1, ..., H6, HEADER, HGROUP, LI, NAV, OL, P, PRE, SECTION, TABLE, UL

Elementos de linha

- A caixa não gera uma nova linha, seu tamanho é definido pelo conteúdo
- Não deve conter elementos de bloco
- Não respeita as propriedades de altura e largura
- As margens e espaçamento interno não possuem o mesmo efeito
 - Comportamento diferente entre horizontal e vertical
- Elementos HTML de linha:
 - A, ABBR, B, BR, BUTTON, CITE, CODE, EM, I, IMG, INPUT, LABEL, MARK, Q, S, SELECT, SMALL, SPAN, STRONG, U, VAR

Módulo IV - Aprendendo CSS

Unidades de Medida

Medidas em CSS

- CSS utiliza de diferentes unidades de medidas para expressar um tamanho
- Há diversas propriedades que o valor é um “tamanho”
- O valor destas propriedades são expressos em um unidade
 - Ex: 15px, 20cm, 2em
- O número é seguido da unidade sem espaço
- O valor 0 não precisa de unidade
- Algumas propriedades permitem medidas negativas
- Há 2 tipos de unidades: absolutas e relativas

Medidas Absolutas (para material impresso)

Unidade	Descrição
cm	centímetros
mm	milímetros
in	polegadas (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	pontos (1pt = 1/72 of 1in)

Medidas Relativas (para tela)

px - Relativo ao pixel da tela

em - Relativo ao tamanho da fonte do elemento

rem - Relativo ao tamanho da fonte do elemento raiz

% (porcentagem) - relativo ao elemento pai

Sobre as unidades de medidas

https://www.w3.org/Style/Examples/007/units.pt_BR.html

	Recomendado	Uso ocasional	Não recomendado
Tela	em, px, %	ex	pt, cm, mm, in, pc
Impresso	em, cm, mm, in, pt, pc, %	px, ex	

Módulo IV - Aprendendo CSS

Tamanho

Altura e Largura

- Os elementos de blocos podem ter sua altura e largura definidos
- Use as propriedades **height** e **width**
- Os valores para estas propriedades podem ser:
 - **auto** - o navegador define qual será o tamanho
 - **tamanho** - px, %, ...
 - **initial** - Define o tamanho padrão
 - **inherit** - Define o mesmo dos pais
- Exemplo:
 - `div { width: 70%; height: 50px; }`

Largura/altura máxima

- Define o tamanho máximo de um elemento
- Este será o tamanho apresentado se a interface for maior do que o tamanho
- Caso a tela seja menor, o elemento terá um tamanho menor
- Exemplo:
 - `div { max-width: 100px; max-height: 50px; }`

Largura/altura mínima

- Define o tamanho mínimo de um elemento
- Se o conteúdo é menor do que o tamanho mínimo então o tamanho será aplicado, caso contrário não se notará o efeito desta propriedade
- Exemplo:
 - `div { min-width: 100px; min-height: 50px; }`

Módulo IV - Aprendendo CSS

Margin e Padding

Margem

- Cria espaço entre os elementos (bordas)
- A propriedade utilizada para definir a margem é **margin**
 - Ex: `p { margin: 10px; }`
- Os valores permitidos para margem são:
 - **auto** - utilizado para centralizar um elemento
 - **tamanho** - px, em, %, ...
 - **inherit** - Mesmo valor do elemento pai
- Obs: As margens podem ter valores negativos

Diferentes margens

- A propriedade `margem` pode receber 4 valores:
 - `p { margin: topo direita base esquerda; }`
- A propriedade `margem` pode receber 3 valores:
 - `p { margin: topo direita-esquerda base }`
- A propriedade `margem` pode receber 2 valores:
 - `p { margin: topo-base direita-esquerda }`
- A propriedade `margem` pode receber 1 valor:
 - `p { margin: topo-base-direita-esquerda }`

Alterando apenas uma margem

- É possível definir apenas uma das margens de um objeto
- Topo:
 - `margin-top`
- Base:
 - `margin-bottom`
- Esquerda:
 - `margin-left`
- Direita
 - `margin-right`

Junção das margens

- Ocorre apenas entre as margens inferior e superior de 2 elementos
- O tamanho da maior margem é assumida como distância

Padding

- Cria espaço entre o conteúdo e a borda do elemento
- Os valores permitidos para a propriedade *padding* são:
 - **tamanho** - px, em, %, ...
 - **inherit** - Mesmo valor do elemento pai
- Diferente das margens, o *padding* não aceita valores negativo

Diferentes espaçamentos

- A propriedade padding pode receber 4 valores:
 - `p { padding: topo direita base esquerda; }`
- A propriedade padding pode receber 3 valores:
 - `p { padding: topo direita-esquerda base }`
- A propriedade padding pode receber 2 valores:
 - `p { padding: topo-base direita-esquerda }`
- A propriedade padding pode receber 1 valor:
 - `p { padding: topo-base-direita-esquerda }`

Alterando apenas um espaçamento interno

- É possível definir apenas uma das distâncias de um objeto
- Topo:
 - `padding-top`
- Base:
 - `padding-bottom`
- Esquerda:
 - `padding-left`
- Direita
 - `padding-right`

Módulo IV - Aprendendo CSS

Bordas

Bordas

- A borda é definida por 3 características:
 - Estilo
 - Tamanho
 - Cor
- Para definir a mesma borda em todos os lados use:
 - `border: tamanho estilo cor;`

Estilo

- A propriedade **border-style** define o estilo da borda
- Os valores aceitos são:
 - dotted, dashed, solid, double, groove, ridge, inset, outset, none, hidden
- Exemplo:
 - `p { border-style: double; }`
- Para definir múltiplos estilos:
 - `border-style: topo direita base esquerda`
 - `border-style: topo direita-esquerda base`
 - `border-style: topo-base direita-esquerda`

Estilo

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

A mixed border.

Tamanho

- A propriedade **border-width** define o tamanho da borda
- O valor da propriedade é definido em alguma unidade de medida
- Exemplo:
 - `p { border-width: 5px; }`
- Para definir múltiplos tamanhos:
 - `border-width: topo direita base esquerda`
 - `border-width: topo direita-esquerda base`
 - `border-width: topo-base direita-esquerda`

Cor da borda

- A propriedade **border-color** define a cor da borda
- Existem diferentes maneiras de definir uma cor (vamos usar o nome)
- Exemplo:
 - `p { border-color: red; }`
- Para definir múltiplas cores:
 - `border-color: topo direita base esquerda`
 - `border-color: topo direita-esquerda base`
 - `border-color: topo-base direita-esquerda`

Diferentes bordas

```
p {  
  
  border-top-style: dotted;  
  
  border-right-style: solid;  
  
  border-bottom-style: dotted;  
  
  border-left-style: solid;  
  
}
```

```
p {  
  
  border-top-width: 1px;  
  
  border-right-width: 0.2em;  
  
  border-bottom-width: 2px;  
  
  border-left-width: 4px;  
  
}
```

```
p {  
  
  border-top-color: black;  
  
  border-right-color: green;  
  
  border-bottom-color: blue;  
  
  border-left-color: red;  
  
}
```

Diferentes bordas

- É possível definir as 3 propriedades para cada uma das bordas
 - border-left
 - border-right
 - border-top
 - border-bottom

Módulo IV - Aprendendo CSS

Cores RGB

Sistema de cores

- Nome das cores
- RGB / RBGA
- HEX
- HSL / HSLA

Nome das cores

- 140 cores nomeadas
- https://www.w3schools.com/colors/colors_names.asp
- https://www.w3schools.com/colors/colors_groups.asp

RGB

- `rgb(RED, GREEN, BLUE)`
- Cada cor corresponde um número entre 0 e 255
 - 0 - Ausência completa da cor
 - 255 - Presença completa da cor
- Exemplos:
 - `rgb(255, 0, 0)` - Vermelho
 - `rgb(0, 255, 0)` - Verde
 - `rgb(255, 255, 0)` - Amarelo
 - `rgb(0, 0, 0)` - Preto
 - `rgb(255, 255, 255)` - Branco

Seletor de cor: https://www.w3schools.com/css/css_colors_rgb.asp

RGBA

- `rgb(RED, GREEN, BLUE, alpha)`
- Especifica a opacidade da cor
- Alpha é o canal de transparência, varia entre 0 e 1:
 - 0 - completamente transparente
 - 1 - Completamente sem transparência

Seletor de cor: https://www.w3schools.com/css/css_colors_rgb.asp

HEX

- O modelo RGB definido em hexadecimal
- Hexadecimal:
 - 16 valores
 - 0, 1, 2, ..., 9, A, B, C, D, E, F
- Utiliza o símbolo # para definir a cor
 - #rrggbb
 - #rgb

Seletor de cor: https://www.w3schools.com/css/css_colors_hex.asp

Módulo IV - Aprendendo CSS

Cores HSL

Sistema de cores

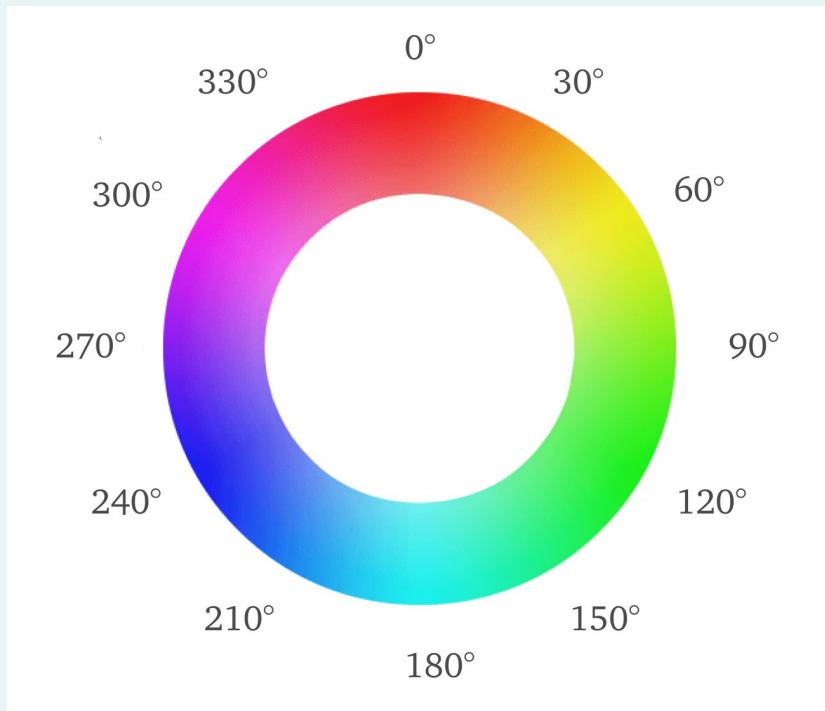
- Nome das cores
- RGB / RBGA
- HEX
- HSL / HSLA

HSL

- *Hue, Saturation, Lightness* (Matiz, Saturação e Luminosidade)
 - Matiz define a cor em uma escala de cores
 - Saturação define a tonalidade de cinza
 - Luminosidade define a quantidade de luz
- `hsl(matiz, saturação, luminosidade)`
 - Matiz: 0 a 360
 - Saturação: 0% a 100% (completamente cinza para completamente a cor)
 - Luminosidade: 0% a 100% (completamente preto para completamente branco)

Seletor de cor: https://www.w3schools.com/css/css_colors_hsl.asp

Matiz



HSLA

- *Hue, Saturation, Lightness, Alpha*
- Inclui o canal de transparência
- `hsl(matiz, saturação, luminosidade, alpha)`
 - Matiz: 0 a 360
 - Saturação: 0% a 100% (completamente cinza para completamente a cor)
 - Luminosidade: 0% a 100% (completamente preto para completamente branco)
 - Alpha: 0 a 1 (completamente transparente para nada transparente)

Seletor de cor: https://www.w3schools.com/css/css_colors_hsl.asp

Definição das cores

- O sistema de cores são correspondentes
 - <https://hslpicker.com/>
- Utilize paletas de cores para criar uma experiência melhor para o usuário
 - <https://coolors.co/palettes/trending>
 - <https://imagecolorpicker.com/>
 - https://www.w3schools.com/colors/colors_palettes.asp

Módulo IV - Aprendendo CSS

Fundo

Backgrounds

- O *background* possui as propriedades
 - Cor
 - Imagem
 - Repetição
 - Posição

Cor

- A propriedade **background-color** define uma cor de fundo
- A cor pode ser definida pelo nome, RGB, HEX e HSL
- Exemplo:
 - `p { background-color: rgb(123, 22, 221); }`
- Também é possível definir a opacidade do elemento
 - Utilize um valor entre 0 e 1 (completamente transparente)
 - `p { opacity: 0.6; }`

Imagem

- É possível atribuir uma imagem de fundo
- A sintaxe utilizada é
 - `background-image: url("pasta/arquivo.jpg");`
- Por padrão a imagem irá se repetir na horizontal e vertical

Repetição

- É possível definir como uma imagem de fundo irá se repetir
- As opções de repetição são:
 - **repeat**: repete em todas as direções
 - **repeat-x**: Horizontalmente
 - **repeat-y**: Verticalmente
 - **no-repeat**: Não repete
- A sintaxe utilizada é
 - `background-repeat: valor;`

Posicionamento

- Principalmente utilizado se a imagem de fundo não se repetir
- A propriedade utilizada é **background-position**
- Os valores para esta propriedade são:
 - **A B** - [left | right | center] [top | center | bottom]
 - **x% y%**
 - **xpos ypos**
 - **inherit**
- Exemplo
 - `p { background-position: right top; }`