Construção de Software para Web

Estilo

Prof. Guilherme Zucatelli
2º Período - 2022

Programação Web

Relembrando…

- **HTML (*Hypertext Markup Language*):** Linguagem principal, responsável pela estruturação dos elementos existentes nas páginas Web.
- **CSS (*Cascading Style Sheets*):** Ferramenta de design, estilo, adotada para descrever/modificar a apresentação dos elementos da página.
- **JS (*JavaScript*):** Linguagem de script para páginas Web.

Elementos de Estilo

- **CSS (*Cascading Style Sheets*):**

O CSS é uma tecnologia que nos permite especificar o **estilo dos elementos visuais** (por vezes chamada de interface do usuário, interface do cliente) de uma aplicação Web.

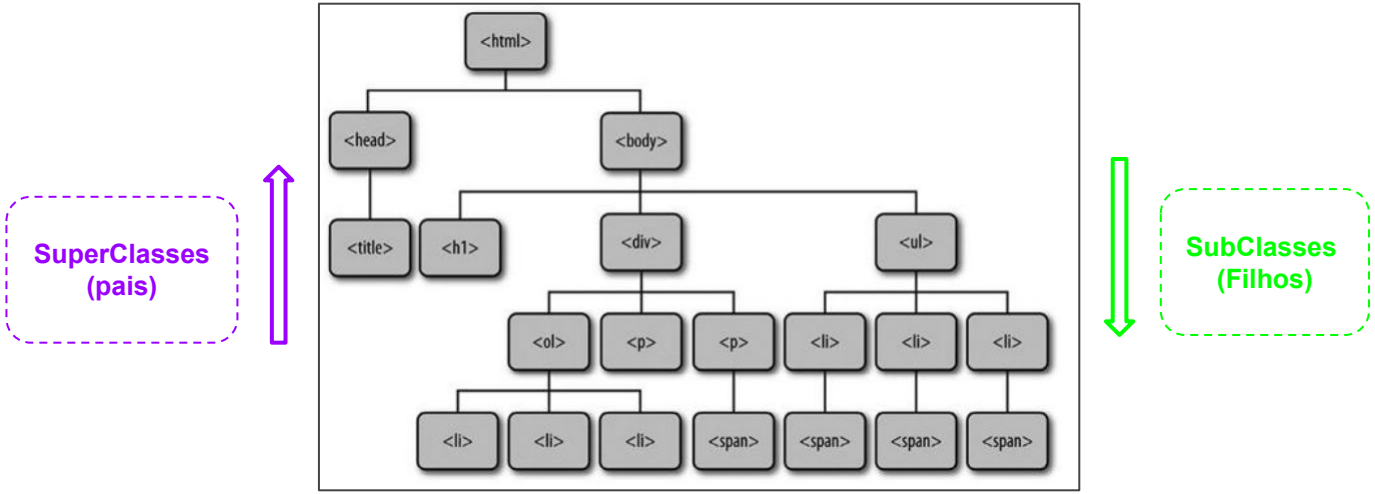
Os arquivos de CSS são organizados a partir da **estrutura hierárquica** do site por meio da definição das **características das classes** e pelos **fundamentos de herança**.

Elementos de Estilo

- **CSS (*Cascading Style Sheets*):**

O CSS é uma tecnologia que nos permite especificar o **estilo dos elementos visuais** (por vezes chamada de interface do usuário, interface do cliente) de uma aplicação Web.

Os arquivos de CSS são organizados a partir da **estrutura hierárquica** do site por meio da definição das **características das classes** e pelos **fundamentos de herança**.



Elementos de Estilo

Estilizando Elementos Estruturais Básicos:

Para o estudo de nossas configurações de estilo, vamos considerar uma aplicação Web fictícia com alguns elementos estilizados.

Observe o arquivo “index.css” a seguir, com alguns parâmetros de estilo para TAGs:

```
/* Arquivo de Estilo */
body{
  background: lightblue;
  width: 800px;
  margin: auto;
  }
h1{
  color: maroon;
  text-align: center;
  }
p{
  color: gray;
  border: 1px solid gray;
  padding: 10px;
  }

```

Elementos de Estilo

Estilizando Elementos Estruturais Básicos:

Para o estudo de nossas configurações de estilo, vamos considerar uma aplicação Web fictícia com alguns elementos estilizados.

Observe o arquivo “index.css” a seguir, com alguns parâmetros de estilo para TAGs:

```
1 /* Arquivo de Estilo */
2 body{
3   background: lightblue;
4   width: 800px;
5   margin: auto;
6   }
7 h1{
8   color: maroon;
9   text-align: center;
10  }
11 p{
12  color: gray;
13  border: 1px solid gray;
14  padding: 10px;
15  }
16
```

Elementos de Estilo

Estilizando Elementos Estruturais Básicos:

Para o estudo de nossas configurações de estilo, vamos considerar uma aplicação Web fictícia com alguns elementos estilizados.

Observe o arquivo “index.css” a seguir, com alguns parâmetros de estilo para TAGs:

```
1 /* Arquivo de Estilo */
2 body{
3   background: lightblue;
4   width: 800px;
5   margin: auto;
6   }
7 h1{
8   color: maroon;
9   text-align: center;
10  }
11 p{
12  color: gray;
13  border: 1px solid gray;
14  padding: 10px;
15  }
16
```

background: Plano de Fundo
width: Espessura
margin: Margem

Elementos de Estilo

Estilizando Elementos Estruturais Básicos:

Para o estudo de nossas configurações de estilo, vamos considerar uma aplicação Web fictícia com alguns elementos estilizados.

Observe o arquivo “index.css” a seguir, com alguns parâmetros de estilo para TAGs:

```
1 /* Arquivo de Estilo */
2 body{
3   background: lightblue;
4   width: 800px;
5   margin: auto;
6   }
7 h1{
8   color: maroon;
9   text-align: center;
10  }
11 p{
12  color: gray;
13  border: 1px solid gray;
14  padding: 10px;
15  }
16
```

background: Plano de Fundo
width: Espessura
margin: Margem

color: Cor
text-align: Alinhamento Texto

Elementos de Estilo

Estilizando Elementos Estruturais Básicos:

Para o estudo de nossas configurações de estilo, vamos considerar uma aplicação Web fictícia com alguns elementos estilizados.

Observe o arquivo “index.css” a seguir, com alguns parâmetros de estilo para TAGs:

```
1 /* Arquivo de Estilo */
2 body{
3   background: lightblue;
4   width: 800px;
5   margin: auto;
6   }
7 h1{
8   color: maroon;
9   text-align: center;
10  }
11 p{
12  color: gray;
13  border: 1px solid gray;
14  padding: 10px;
15  }
16
```

background: Plano de Fundo
width: Espessura
margin: Margem

color: Cor
text-align: Alinhamento Texto

color: Cor
border: Bordas
padding: Espaçamento

Elementos de Estilo

Link com Folha de Estilo:

Para associar as regras de estilo desejadas à nossa página HTML precisamos **vincular** o arquivo HTML com o estilo definido em nosso arquivo CSS.

Para tanto adicionamos a TAG **<link>** em nosso arquivo HTML com os atributos de especificação:

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

```css
/* Arquivo de Estilo */
body{
  background: lightblue;
  width: 800px;
  margin: auto;
  }
h1{
  color: maroon;
  text-align: center;
  }
p{
  color: gray;
  border: 1px solid gray;
  padding: 10px;
  }
```

index.css

```html
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="index.css">
    <title>LifeStyle Capixaba</title>
  </head>
  <body>
  </body>
</html>
```

Elementos de Estilo

Inclusão de Estilo Básico em TAGs:

As regras de estilo na linguagem CSS (*Cascading Style Sheet*) são definidas em cascata. Isto significa que as regras são sobrepostas umas sobre as outras, dependendo de suas definições.

Além disso, é importante observar a hierarquia dos elementos presentes em um arquivo HTML, pois regras de estilo podem ser herdadas do elemento pai.

Atributo	Descrição
background:	Definição do plano de fundo
color:	Definição de cores
width:	Largura do elemento
margin:	Margem: tipo de espaçamento para outros elementos
left-/rigth-margin:	Margens à esquerda e à direita
text-align:	Alinhamento do texto: *left, right, center, justify*
font-family:	Família de fontes: serif, sans-serif, monospace, cursive, fantasy

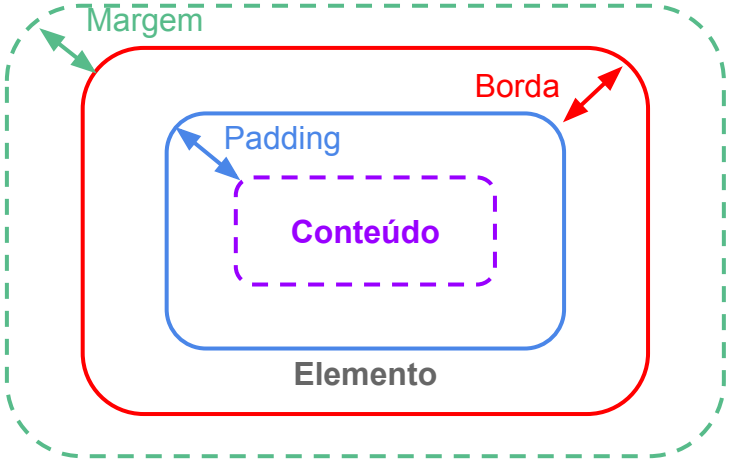
[Tabela de Cores]

Elementos de Estilo

Espaçamento em Elementos de Bloco:

Alguns dos mais importantes elementos de construção de aplicações Web são definidos por uma estrutura em blocos. Na prática, esses elementos são normalmente incluídos na página em uma nova linha.

Alguns exemplos de **elementos de bloco** são: **<p>**, **<h1>-<h6>**, **<ol>**, **<ul>**, **<dl>**, **<hr>, <img>**, ** **



Atributo	Descrição
margin:	Espaçamento de Margem
left-margin:	Margens à esquerda
right-margin:	Margens à direita
padding:	Espaçamento *Padding*
border:	Espaçamento de Borda
border-color:	Cor da Borda
border-width:	Largura da Borda
border-style:	Estilo da Borda: *dashed, solid,* …

Elementos de Estilo

Inclusão Básica de Estilo em Classes:

É muito comum que o atributo class=“” seja adotado para a definição de regras de estilo. Nesse caso, o estilo deve ser definido como .nome_da_classe, como no exemplo a seguir:

```
20  a {
21    padding: 15px;
22  }
23
24  .header-conteudo {
25    width: 700px;
26    margin: auto;
27  }
28
29  .main-conteudo {
30    width: 700px;
31    margin-left: auto;
32    margin-right: auto;
33    text-align: justify;
34    font-family: cursive;
35  }
```

Elementos de Estilo

Inclusão Básica de Estilo em Classes:

É muito comum que o atributo **class=“”** seja adotado para a definição de regras de estilo. Nesse caso, o estilo deve ser definido como **.nome_da_classe**, como no exemplo a seguir:

```
20  a {
21    padding: 15px;
22  }
23
24  .header-conteudo {
25    width: 700px;
26    margin: auto;
27  }
28
29  .main-conteudo {
30    width: 700px;
31    margin-left: auto;
32    margin-right: auto;
33    text-align: justify;
34    font-family: cursive;
35  }
```

Uma das principais importâncias de **regras definidas em classes** pode ser observado para o elemento genérico ** **.

Como esse elemento é genérico, podemos definir várias seções diferentes de nossa aplicação com regras de estilo diversificadas.

Nota: Caso existam classes com mesmo nome com comportamentos diferentes, deve-se definir a classe com base na TAG a ser adotada, como:

```
h2.primeiro-paragrafo{...
    }

h3.primeiro-paragrafo{...
    }
```

Elementos de Estilo

Atividade:

Com base em seus conhecimentos, estilize a página do blog “LifeStyle Capixaba”.

(a) Utilize as organizações de conteúdo da TAG <body>: <header>, <main> e <footer>.

(b) Adicione elementos visuais, **<img>** e **<video>**

(c) Adote a TAG ** ** para definir alinhamento central para o conteúdo no site com o atributos *width*.

(d) Defina regras de estilo para suas TAGs e classes diversas.

Exemplo Prático

Atividade

Elementos de Estilo

Inclusão de Estilo em Textos:

- Google Fonts API [Link]: Variedade de Fontes (e Ícones) com licença gratuita que podem ser adotadas para estilização das fontes de sua aplicação Web (font-family).

```
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=FAMILY_NAME">
```

Elementos de Estilo

Inclusão de Estilo em Textos:

- Google Fonts API [Link]: Variedade de Fontes (e Ícones) com licença gratuita que podem ser adotadas para estilização das fontes de sua aplicação Web (font-family).

```
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=FAMILY_NAME">
```

- Atributos Adicionais:

Atributo	Descrição
text-align:	**Alinhamento do texto:** *left, right, center, justify*
font-family:	**Família de fontes:** serif, sans-serif, monospace, cursive, fantasy **+ Google API**
font-size:	Tamanho da fonte
font-style:	**Estilo Fonte:** normal, italic, oblique
font-weight:	**“Peso”:** bold (negrito), bolder, valor (100-900)

Elementos de Estilo

Inclusão de Estilo em Textos:

- Google Fonts API [Link]: Variedade de Fontes (e Ícones) com licença gratuita que podem ser adotadas para estilização das fontes de sua aplicação Web (font-family).

```
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=FAMILY_NAME">
```

- Atributos Adicionais:

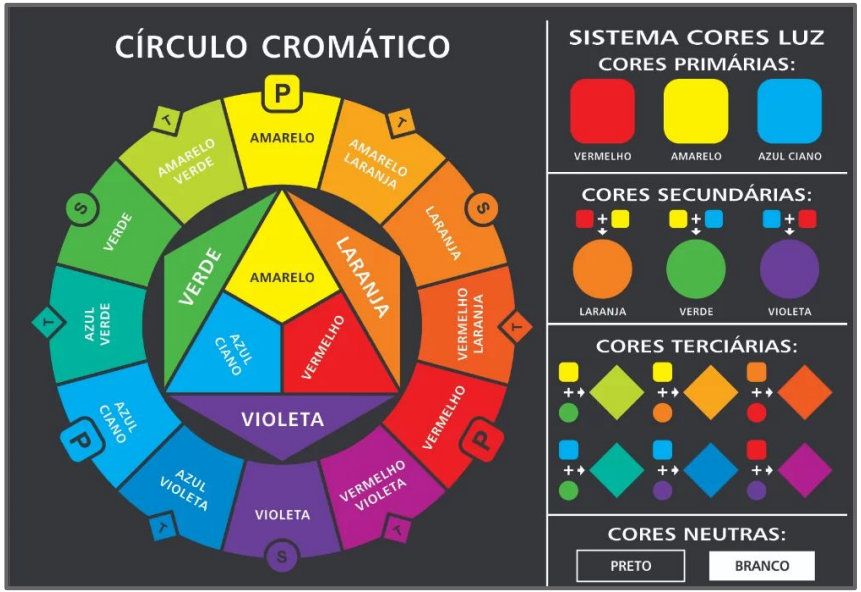
Atributo	Descrição
text-align:	**Alinhamento do texto:** *left, right, center, justify*
font-family:	**Família de fontes:** serif, sans-serif, monospace, cursive, fantasy **+ Google API**
font-size:	Tamanho da fonte
font-style:	**Estilo Fonte:** normal, italic, oblique
font-weight:	**“Peso”:** bold (negrito), bolder, valor (100-900)

- **BackUp:** Pode-se definir múltiplas fontes para um determinado elemento/classe, estipulando uma prioridade para o caso de alguma fonte não estar acessível.

Elementos de Estilo

Definindo Cores em CSS:

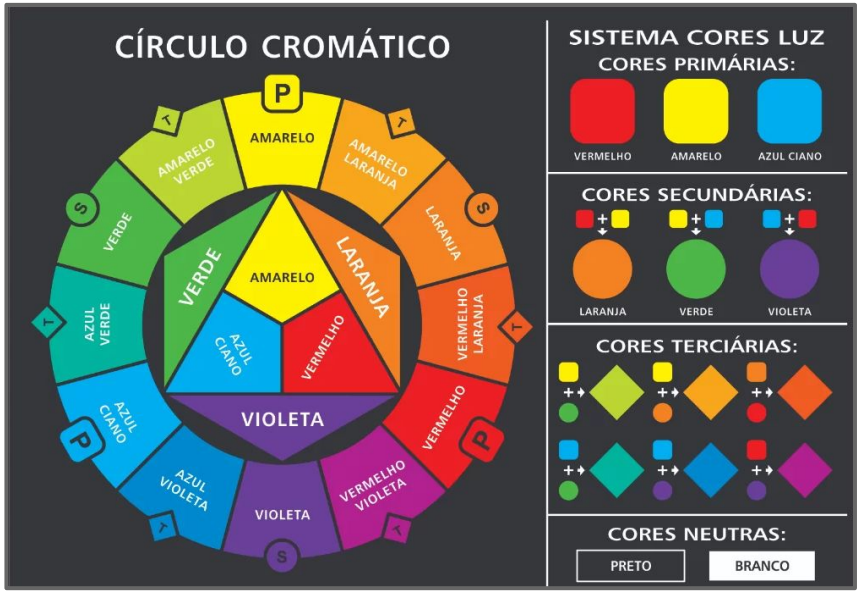
- Teoria das Cores:



Elementos de Estilo

Definindo Cores em CSS:

- Teoria das Cores:

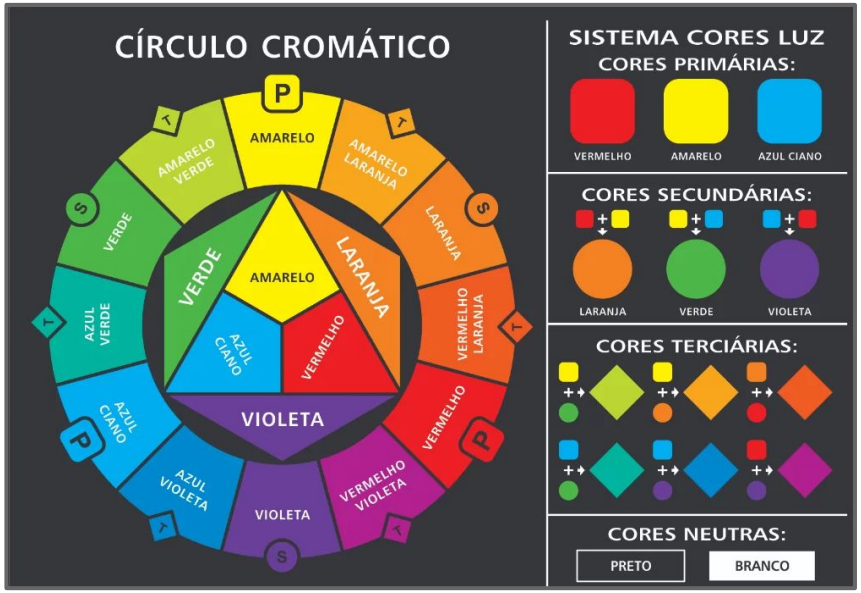


RGB x CMYK

Elementos de Estilo

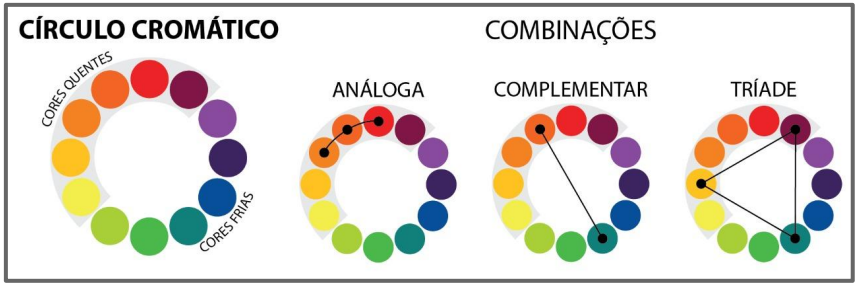
Definindo Cores em CSS:

- Teoria das Cores:



RGB x CMYK

- Combinações e Paletas:



A escolha das cores é essencial para a definição de uma identidade gráfica do seu site.

Para tanto, é muito comum adotar uma seleção de cores que são **harmônicas entre si**, denominada **Paleta de Cores**.

Veja algumas opções: https://colorhunt.co/

Elementos de Estilo

Definindo Cores em CSS:

- RGB:

Uma forma comum de definir cores é pela mistura das componentes Red-Green-Blue.

Em CSS (e inúmeras outras aplicações) as partes de cada cor RGB são definidas considerando uma escala de 0-255 (total de $256=2^8=16^2$ opções).

```
color: rgb(0, 0, 255);
color: rgb(150, 0, 0);
color: rgb(15, 122, 68);
```

Elementos de Estilo

Definindo Cores em CSS:

- RGB:

Uma forma comum de definir cores é pela mistura das componentes Red-Green-Blue.

Em CSS (e inúmeras outras aplicações) as partes de cada cor RGB são definidas considerando uma escala de 0-255 (total de $256=2^8=16^2$ opções).

```
color: rgb(0, 0, 255);
color: rgb(150, 0, 0);
color: rgb(15, 122, 68);
```

- Escala Hexadecimal (Hex):

A escala hexadecimal adota a base 16 para representação dos números.

Um número **XY** na base 10: $X*10^1+Y*10^0$
Um número **XY** na base 16: $X*16^1+Y*16^0$

Algarismos da **Base Hex**: 0,1,2,...,9,A,B,C,D,E,F

Elementos de Estilo

Definindo Cores em CSS:

- RGB:

Uma forma comum de definir cores é pela mistura das componentes Red-Green-Blue.

Em CSS (e inúmeras outras aplicações) as partes de cada cor RGB são definidas considerando uma escala de 0-255 (total de $256=2^8=16^2$ opções).

- Escala Hexadecimal (Hex):

A escala hexadecimal adota a base 16 para representação dos números.

Um número XY na base 10: $X*10^1+Y*10^0$
Um número XY na base 16: $X*16^1+Y*16^0$

Algarismos da **Base Hex**: 0,1,2,...,9,A,B,C,D,E,F

```
color: rgb(0, 0, 255);
color: rgb(150, 0, 0);
color: rgb(15, 122, 68);
```

→

```
color: #0000FF;
color: #960000;
color: #0F7A44;
```

Elementos de Estilo

Definindo Cores em CSS:

- RGB:

Uma forma comum de definir cores é pela mistura das componentes Red-Green-Blue.

Em CSS (e inúmeras outras aplicações) as partes de cada cor RGB são definidas considerando uma escala de 0-255 (total de $256=2^8=16^2$ opções).

- Escala Hexadecimal (Hex):

A escala hexadecimal adota a base 16 para representação dos números.

Um número **XY** na base 10: $X*10^1+Y*10^0$
Um número **XY** na base 16: $X*16^1+Y*16^0$

Algarismos da **Base Hex**: 0,1,2,...,9,A,B,C,D,E,F

```
color: rgb(0, 0, 255);
color: rgb(150, 0, 0);
color: rgb(15, 122, 68);
```

→

```
color: #0000FF;
color: #960000;
color: #0F7A44;
```

Sugestão para escolha de Cores: https://www.webfx.com/web-design/color-picker/

Fator translúcido (**alpha**): Use rgba(15, 122, 68, 0.3) e #0F7A44CA;

Elementos de Estilo

Variáveis CSS e Identificador (id):

Definir **variáveis** é muito importante para especificações pertinentes da página.

As variáveis podem ser declaradas dentro de um **escopo de classe**, com acesso a todas as suas subclasses, ou em todo o documento (root).

Para adotá-las: var(nome_variavel);

```
5   :root{
6     --cor_paleta1: #69BEF5;
7   }
8
9   body {
10    background: white;
11    font-family: cursive;
12    --cor_paleta2: #E669F5;
13  }
14
```

index.css

Elementos de Estilo

Variáveis CSS e Identificador (id):

Definir **variáveis** é muito importante para especificações pertinentes da página.

As variáveis podem ser declaradas dentro de um **escopo de classe**, com acesso a todas as suas subclasses, ou em todo o documento (root).

Para adotá-las: var(nome_variavel);

```
5   :root{
6     --cor_paleta1: #69BEF5;
7   }
8
9   body {
10    background: white;
11    font-family: cursive;
12    --cor_paleta2: #E669F5;
13  }
14
```

index.css

- Identificador (id):

Além de classes, os elementos de HTML podem ser identificados por “id”s.

Cada elemento pode possuir um **único id**, o qual apresenta **prioridade sobre as regras de classes**.

Para definições de regras de estilo para identificadores adota-se a **sintaxe com #**, exemplificado a seguir:

```
54  #primeiro_paragrafo{
55    text-align: justify;
56    color: #78F569;
57  }
```

Elementos de Estilo

Atividade:

Com base em seus conhecimentos, estilize a página do blog “LifeStyle Capixaba”.

(a) Utilize uma família de fontes (Google API) específica para o nome do site.

(b) Defina diferentes cores (paletas) que julgue interessantes para o seu site **em variáveis.**

(c) Insira outras informações e adote os identificadores ***ID*** para compor a organização do seu site.

Atividade

Elementos de Estilo

Pseudo-Classes:

Pseudo-classes são adotados para definir um **estado especial** de algum elemento.

Como exemplo, pode-se citar:

- Estilo de um elemento quando se passa o mouse sobre ele
- Estilo de links que já foram visitados e links não visitados
- Estilo de elementos que estão em foco (como caixas de texto)

A sintaxe de uma pseudo-classe é definida da seguinte forma:

```
seletor:pseudo-classe{
        propriedade: valor;
        }
```

Elementos de Estilo

Pseudo-Classes em <a>:

Pseudo-classes são comumente adotadas para definir o estado de elementos âncoras **<a>** que carregam como um link no atributo 'href'.

Existem quatro pseudo-classes básicas para a definição do estado de links em aplicações Web:

- **link:** define estilos de todos os links não-visitados, como cor
- **visited:** define estilo dos links visitados (info sensível)
- **hover:** define comportamento quando mouse/cursor está em cima do link
- **active:** define comportamento do link ao ser selecionado pelo mouse.

Elementos de Estilo

Pseudo-Classes em <a>:

Pseudo-classes são comumente adotadas para definir o estado de elementos âncoras **<a>** que carregam como um link no atributo 'href'.

Existem quatro pseudo-classes básicas para a definição do estado de links em aplicações Web:

- **link:** define estilos de todos os links não-visitados, como cor
- **visited:** define estilo dos links visitados (info sensível)
- **hover:** define comportamento quando mouse/cursor está em cima do link
- **active:** define comportamento do link ao ser selecionado pelo mouse.

Pseudo-Classes em classes HTML:

Também pode-se combinar as regras de pseudo-classes especificamente para uma TAG.CLASSE de interesse. Por exemplo, se quiséssemos aplicar um **efeito de *hover*** apenas às **âncoras** pertencentes à **classe principal**:

```
a:hover{
        propriedade: valor;
        }
```



```
a.principal:hover{
        propriedade: valor;
        }
```

Elementos de Estilo

Display:

A propriedade de Display é uma das mais importantes para o controle de *layout* na estilização de uma página. Essa propriedade define se e como um determinado elemento irá aparecer em uma página.

Todo elemento HTML possui um valor de display ***default*** que na maioria das vezes são **block** ou **inline**.

Elementos de Estilo

Display:

A propriedade de Display é uma das mais importantes para o controle de *layout* na estilização de uma página. Essa propriedade define se e como um determinado elemento irá aparecer em uma página.

Todo elemento HTML possui um valor de display ***default*** que na maioria das vezes são **block** ou **inline.**

display: block **:** Um elemento de bloco sempre inicia em uma nova linha e preenche toda a largura (width) disponível. São exemplos de elementos de bloco: , <h1>-<h6>, <p>, <header>, <footer>, <section>, …

display: inline **:** Um elemento de linha não inicia em uma nova linha e preenche o espaço apenas com o necessário, ou seja, não se adapta a toda a largura (width) disponível. São exemplos de elementos de linha: , <a>, <img>, …

Elementos de Estilo

Modificando o Atributo Display:

Como comentado, todo elemento possui um valor de display default. No entanto, podemos sobrescrever esses valores para alterar o comportamento de determinados elementos.

Alguns exemplos práticos:

display: none : Quando associamos o valor "none" a um elemento, esse elemento simplesmente deixa de aparecer no navegador. Isso é especialmente importante quando adotamos JavaScript para alterar atributos, fazendo com que elementos possam desaparecer.

li com **display: inline** : Podemos escrever itens de listas com comportamento *inline,* por exemplo, para criar links de navegação em uma mesma linha, como os adotados no header de páginas web.

Elementos de Estilo

Display inline-block:
Comparado ao display: inline, a maior diferença é que display: inline-block permite a alteração de largura e altura do elemento.

Além disso, para display: inline-block as margens e paddings superiores e inferiores são respeitados.

Comparado ao display: block, a maior diferença é que display: inline-block não adiciona uma quebra de linha após o elemento. Então o elemento pode **se posicionar ao lado de outros elementos**.

Elementos de Estilo

Atividade:

Com base nos elementos apresentados em sala, estilize a página do blog “LifeStyle Capixaba”.

(a) Utilize as pseudo-classes para estilizar o comportamento dos elementos da página: altere cor e inclua modificações às âncoras presentes no site.

(b) Busque formas de criar "botões" nos elementos de navegação. Mude a cor dos botões ao pressioná-los com o mouse.

(c) Selecione outras imagens (de 2 a 5) que sejam representativas do estado do Espírito Santo e inclua essas imagens no site de forma que fiquem ordenadas lado a lado. Inclua pseudo-classes nas imagens em questão.

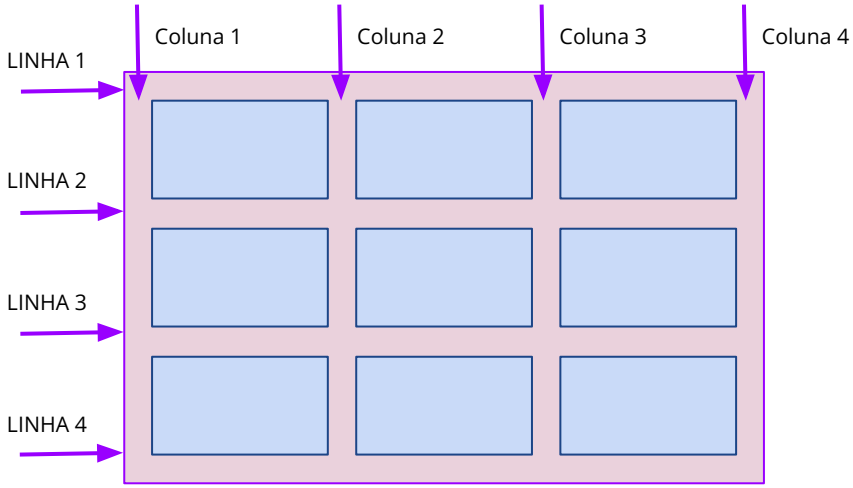
(d) Pesquise por outras formas de organizar elementos em um site.

Atividade

Elementos de Estilo

Estilo Grid:

Elementos HTML podem ser transformados em elementos matriciais pela utilização das regras de estilo de *display*, com a definição do atributo **“display: grid”**;

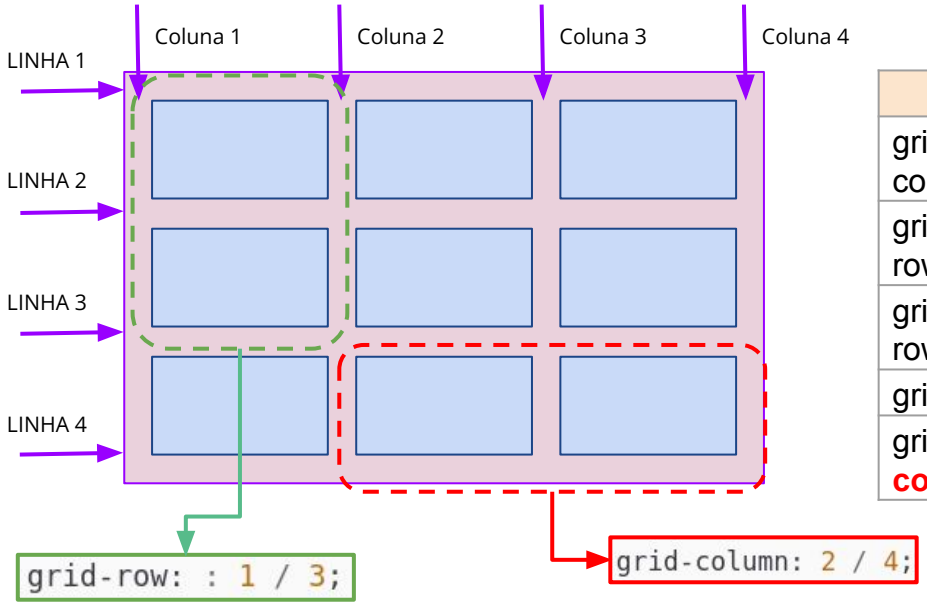


Atributo	Descrição
grid-template-columns	Define espaçamento e número de colunas. Exemplo: grid-template-columns: 50px 100px 10%;
grid-template-rows	Define espaçamento e número de linhas. Exemplo: grid-template-rows: 2fr 1fr 30px;
grid- column / row-gap	Espaçamento entre colunas/linhas
grid-gap	Define espaçamento linha e coluna
grid-**colum**/**row**	Integra elementos do grid, definidos pelo início de fim da coluna/linha. grid-row: 1 / 4;;

Elementos de Estilo

Estilo Grid:

Elementos HTML podem ser transformados em elementos matriciais pela utilização das regras de estilo de *display*, com a definição do atributo **“display: grid”**;



Atributo	Descrição
grid-template-columns	Define espaçamento e número de colunas. Exemplo: grid-template-columns: 50px 100px 10%;
grid-template-rows	Define espaçamento e número de linhas. Exemplo: grid-template-rows: 2fr 1fr 30px;
grid- column / row-gap	Espaçamento entre colunas/linhas
grid-gap	Define espaçamento linha e coluna
grid-**colum**/**row**	Integra elementos do grid, definidos pelo início de fim da coluna/linha. grid-row: 1 / 4;

Elementos de Estilo

Posicionamento em Grid:

O posicionamento dos elementos **dentro de uma malha *grid*** é muito interessante para a estrutura de estilo geral de uma página web.

Por vezes queremos orientar os elementos dentro da estrutura *grid* de formas diversas. Para tanto, alguns atributos de estilo podem ser utilizados:

Atributo	Descrição
align- and justify-items:	Define a orientação de alinhamento de **todos os elementos** dentro da **estrutura *grid***.
justify-self:	Define a orientação de justificativa dos elementos. Exemplo: justify-self: start | center | end;
align-self:	Define a orientação de alinhamento dos elementos. Exemplo: align-self: start | center | end;

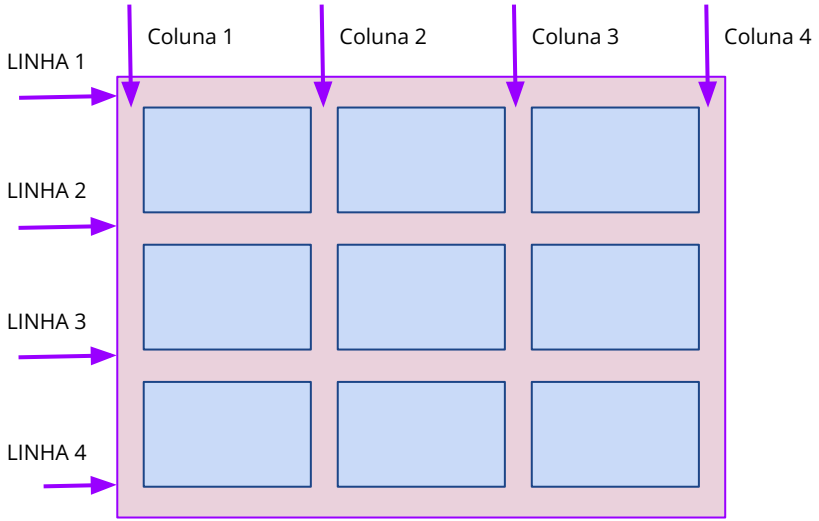
Elementos de Estilo

Definição de Áreas em Grid:

Uma forma interessante de identificar os elementos de uma estrutura *grid* é por meio da nomeação de suas regiões/áreas.

A identificação pode ser feita por meio do atributo ***"grid-template-areas"*** e o posicionamento com **"grid-area".**

```
grid-template-areas:
    "header header header"
    "tadvert tcontent tcontent"
    "badvert bcontent bcontent"
    "footer footer footer"
```



Atributo	Descrição
grid-template-areas: (no bloco grid)	Define a identificação das áreas de um grid.
grid-area: (no elemento)	Define o posicionamento do elemento com base na nomenclatura ou índices do grid. **Ex.:** grid-area: header

Elementos de Estilo

@media query:

Uma das ferramentas introduzidas pelo CSS3 foi a capacidade de alterar certas regras de estilo de acordo com algumas condições.

Essa estratégia é essencial no contexto atual de **multi-devices** com diferentes tamanhos de tela, mas também é adotada para uma melhor organização estrutural do site a depender do tamanho da janela utilizada pelo usuário.

Elementos de Estilo

@media query:

Uma das ferramentas introduzidas pelo CSS3 foi a capacidade de alterar certas regras de estilo de acordo com algumas condições.

Essa estratégia é essencial no contexto atual de **multi-devices** com diferentes tamanhos de tela, mas também é adotada para uma melhor organização estrutural do site a depender do tamanho da janela utilizada pelo usuário.

Ex.:

```
@media (max-width: 700px) {
                body {
                  background-color: lightblue;
                }
        }
```

Mobile First: Desenvolver sites para mobile antes de desktop é fundamental para garantir a velocidade nesses aparelhos. Normalmente isso se traduz em uma regra de tamanho de tela que seja superior aos dispositivos smartphones.

Elementos de Estilo

(Para Pesquisar) Display Flex:

Uma outra forma de organizar os elementos em uma aplicação web é por meio do atributo “display: flex”, de elemento flexível.

Essa estratégia é muito semelhante ao grid visto anteriormente. Alguns atributos importantes de organização dos elementos flex podem ser consultados a seguir:

Atributo	Descrição
flex-direction:	Define o sentido da orientação dos elementos. **Ex.:** flex-direction: row | column | row-reverse | column-reverse;
flex-wrap:	Realiza a “quebra de linha” entre elementos. **Default:** flex-wrap: nowrap;
flex-flow:	Define a *Direction* e Wrap simultaneamente. **Ex.:** flew-flow: row wrap;
justify-content:	Define a justificativa dos elementos flex. **Ex.:** justify-content: flex-start | space-around | center | space-between | …
align-content:	Adotada para posicionamento das “linhas (rows) flex”. **Ex.:** align-content: flex-start | space-around | center | space-between | …

Elementos de Estilo

(Para Pesquisar) Display Flex:

Os elementos dentro de um container flex automaticamente se tornam componentes flex.

Alguns dos principais atributos de edição de elementos flex podem ser observados a seguir:

Atributo	Descrição
order:	Define a ordem de posicionamento dos elementos: **Ex.:** order: 1 | 2 | 3| …
flex-grow/shrink:	Aumenta/diminui elementos flex em comparação aos demais (definido em conjunto). **Ex:** flex-wrap: wrap; flex-grow: 1 | 2 | 3…
flex-basis:	Define o tamanho inicial dos elementos flex antes de aplicar grow/shrink. **Ex.:** flex-basis: 10em
flex:	Aplica os estilos flex -grow, -shrink e -basis simultaneamente: **Ex.:** flex: 0 1 10em;

Elementos de Estilo

Aprofundando os estudos em CSS…

Para aprofundar seus estudos em CSS, existem inúmeros sites de ensino, discussão e disseminação do conhecimento dessa linguagem

Alguns exemplos para estudo:

- www.w3schools.com
- https://css-tricks.com
- https://purecss.io
- *(Game)* https://flexboxfroggy.com

Elementos de Estilo

Atividade:

a) Pratique a estilização de elementos GRID na página “Color Samples”.

b) Com os seus conhecimentos, adapte a estrutura da página LifeStyle Capixaba utilizando grid:

- adicione **alinhamento e justificativa** aos elementos
- defina áreas diversas com **grid-template-areas**, embaralhe o posicionamento dos elementos e defina ocupação de áreas diferentes entre elementos.
- altere as regras de estilo adotando **media query** para considerar diferentes tamanhos de tela dos dispositivos (lembre-se do *mobiles first*).

c) Escolha uma outra disposição dos elementos da página anterior e disponha os elementos de acordo com critérios de estilo FLEX.

Atividade

Prof. Guilherme Zucatelli

PhD on Defense Engineering

e-mail: guilherme.zucatelli@uvv.br