HTML/CSS //ASSUNTO DO MÓDULO

MÓDULO 04 // NÚMERO DO MÓDULO

COMPREENDER A HISTÓRIA E DEFINIÇÃO DO HTML. //NOME ACIMA DOS ICONES DA AULA ABAIXO DO NÚMERO

O que é Html (hitória e definição) //NOME DAS CAPAS DO HYPERTEXTO

O que é HTML//NOME DOS TÍTULOS DOS HYPERTEXTOS

Hypertext Markup Language (HTML), em português, Linguagem de Marcação de Hipertexto, é um conjunto de sinais e códigos aplicados a um texto, ou dados, que **contribui para definir exibições na tela** e de estruturas de dados em um sistema. Ele que dá corpo a página, e organiza as inforamções. Sem ele o navegador não saberia onde iria cada elemento, não exibiria textos ou elementos, ou carregar imagens.

> NÃO É LINGUAGEM DE PROGRAMAÇÃO, POIS NÃO POSSUI PROPRIEDADES COMO OPERADORES LÓGICOS, FUNÇÕES, VARIÁVEIS, ETC.

História

Veio da necessidade de se criar/editar e compartilhar Hipertextos pela internet. Tim Berners Lee que realizou a estrutra da linguágem que possibilitou a comunicação entre Cliente HTTP x Servidor.

Versões

Versão 2, 1995, formalizar suas caracteristicas. Versão 3.2, 1997, correções a add tabelas, applets e texto flutuante ao redor da imagem. Versão 4.01, 1999, compatibilidade entre outras versões, suporte a mídias, folha de stilos. Versão 5, 2015, suporte a áudio e vídeo de alt nível, deixou de utilizar dados só de cache e passou a utilizar banco de dados. Controle de formulários entre outras coisas.

Conclusão

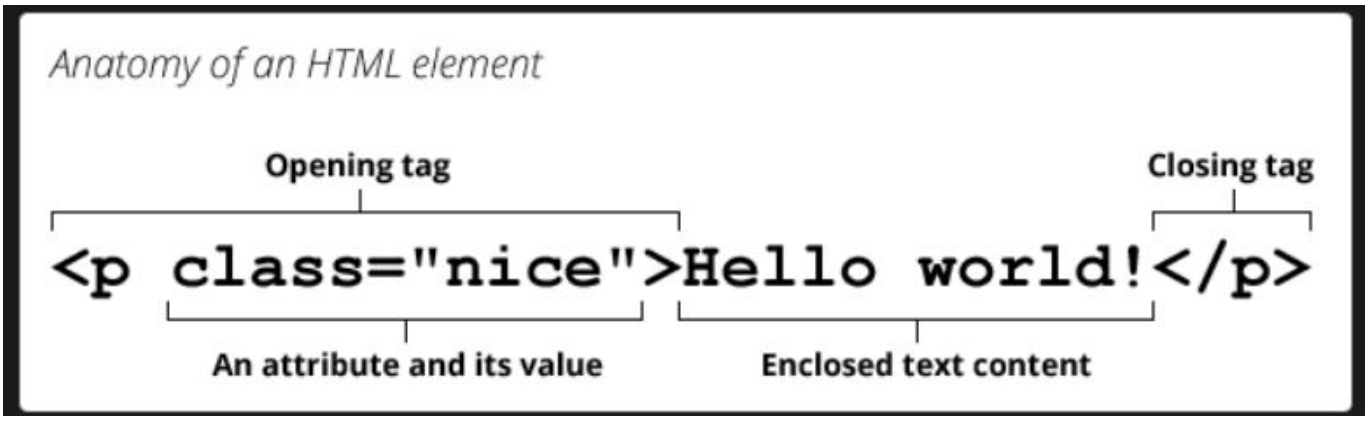
Para o **desenvolvimento web**, **é bastante importante** que você tenha conhecimento básico em HTML. Apesar de já existirem ferramentas que criam o código em HTML, às vezes é preciso fazer adaptações no código. Para isso, você precisa conhecer os comandos HTML.

CONHECER AS PRINCIPAIS TAGS DO HTML

Introdução

Para se contruir um site é necessário uma estrutura mínima base de HTML. TAGS compõe essa estrutura para FORMAÇÃO DE ELEMENTOs que fazem parte do DOM;

Um elemento é parte de uma página web. Em XML e HTML, um elemento pode conter um item de dados, um bloco de texto, uma imagem ou talvez nada. Um elemento típico inclui uma tag de abertura com alguns atributos, o conteúdo de texto incluído e uma tag de fechamento.



O Modelo de Objeto de Documentos (do inglês Document Object Model, DOM) é uma API definida pelo W3C para representar e interagir com qualquer documento HTML ou XML.

O DOM é um modelo de documento carregado pelo navegador. Este documento é representado através de uma árvore de nós, onde cada um destes nós representa uma parte do documento (por ex. um elemento, texto ou comentário).

Introdução as TAGS

As tags são estruturas de linguagem de marcação. Elas contêm instruções que servem para informar a estrutura do site ao navegador e construir elementos da HTML. Quando são utilizadas corretamente, elas contribuem bastante para o SEO é o conjunto de estratégias de otimização para mecanismos de busca. O objetivo é alcançar bom posicionamento orgânico de páginas da web no Google e em outros buscadores" de um site.

Cada tag é formada pelo sinal menor que <, nome do elemento e sinal maior que >, nessa ordem. Dessa forma:

```
 <html>

 existem tags que precisam ser fechadas e outras não. Na maioria dos casos, o fechamento é necessário para definir o início e o fim do documento. Para fechá-la, colocamos a barra / antes do nome do elemento.

 </html>

 Para as tags que não possuem fechamento, é preciso colocar a barra antes do sinal de maior que:

<input/>
```

Atributos

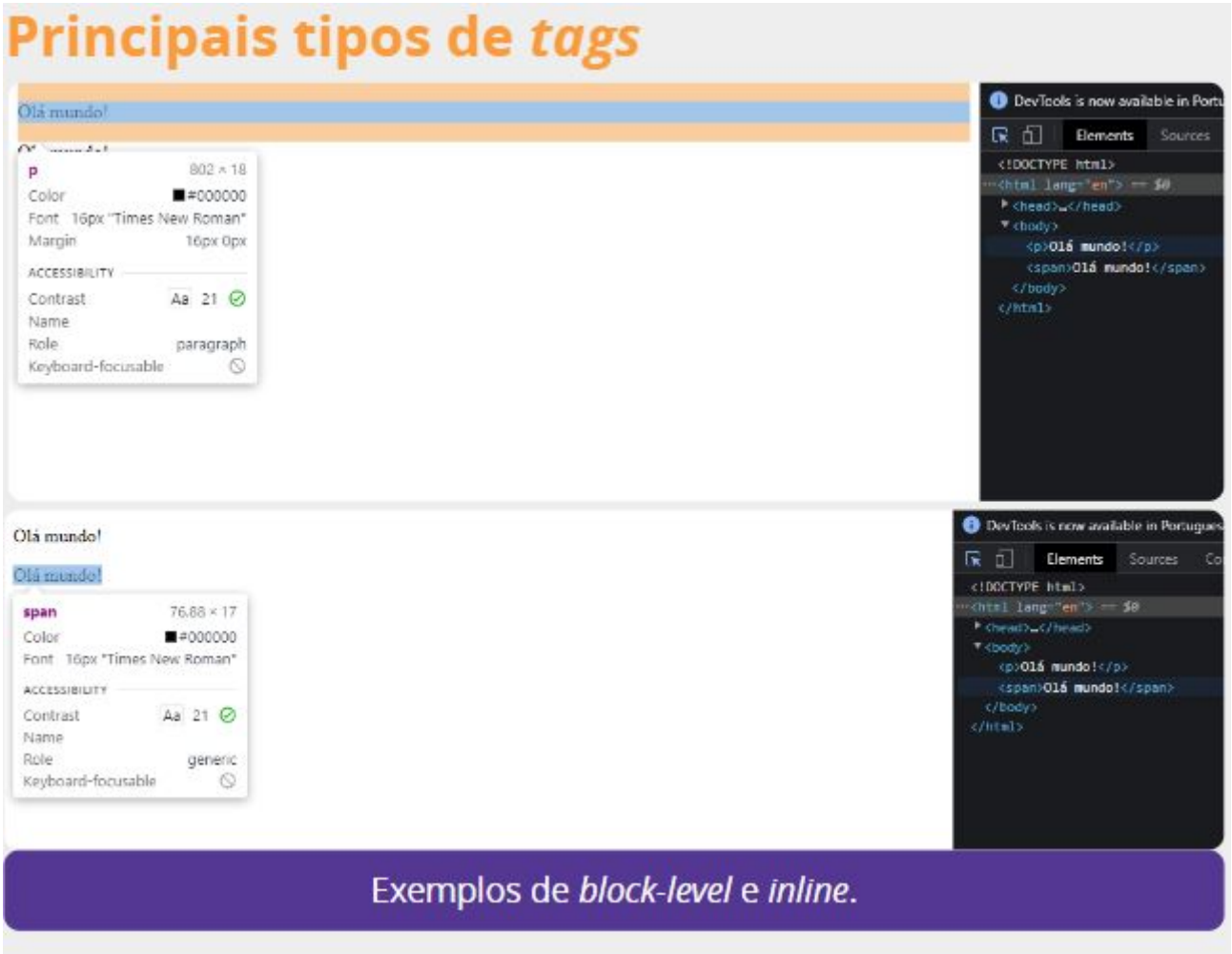
Apesar das tags terem seus próprios atributos, alguns podem ser utilizados em várias tags.

- **class**: atribui uma classe CSS para uma tag. Isso faz com que a CSS e a linguagem Javascript selecionem e acessem elementos específicos do código HTML;

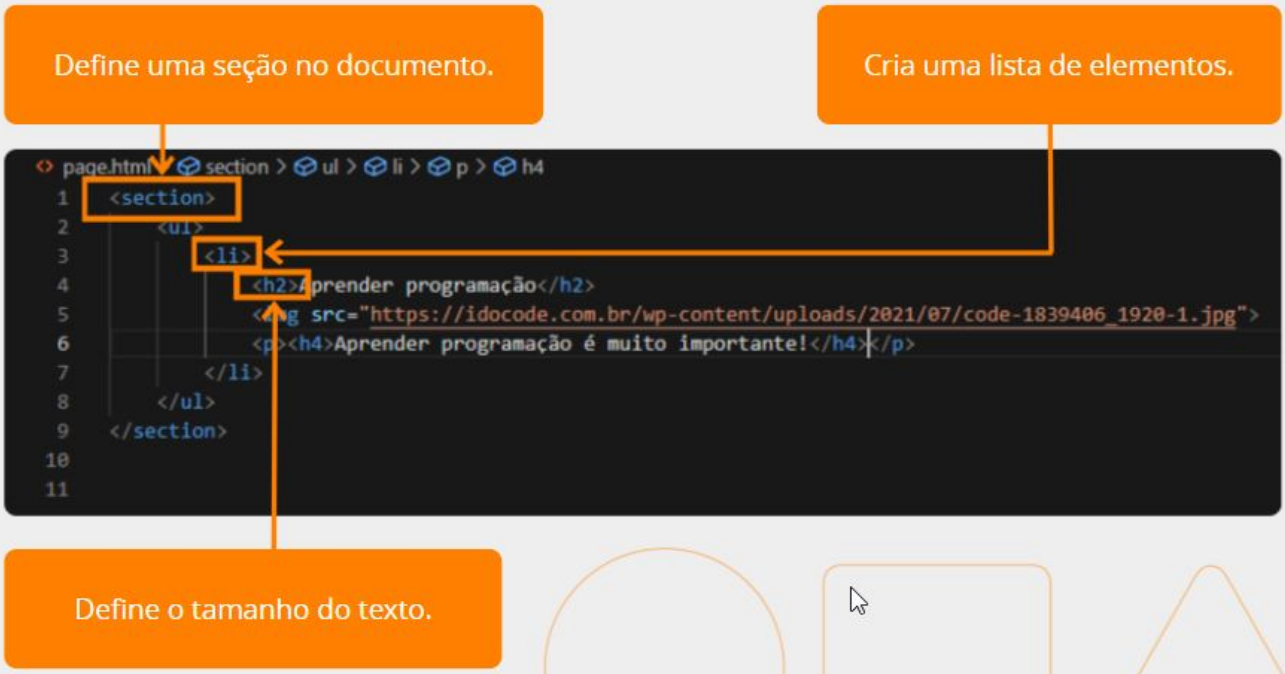
- **id**: atribui um id CSS para uma tag. Seu objetivo é identificar o elemento quando utilizar scripts ou estilizar com CSS;
- **href**: referenciar um URL externo, seja um link ou um arquivo
- **src**: atribui um URL para um conteúdo. Por exemplo, uma imagem ou um arquivo;
- **type**: atribui o tipo do elemento;
- **value**: atribui o valor padrão de um elemento.

Principais tipos de TAGS

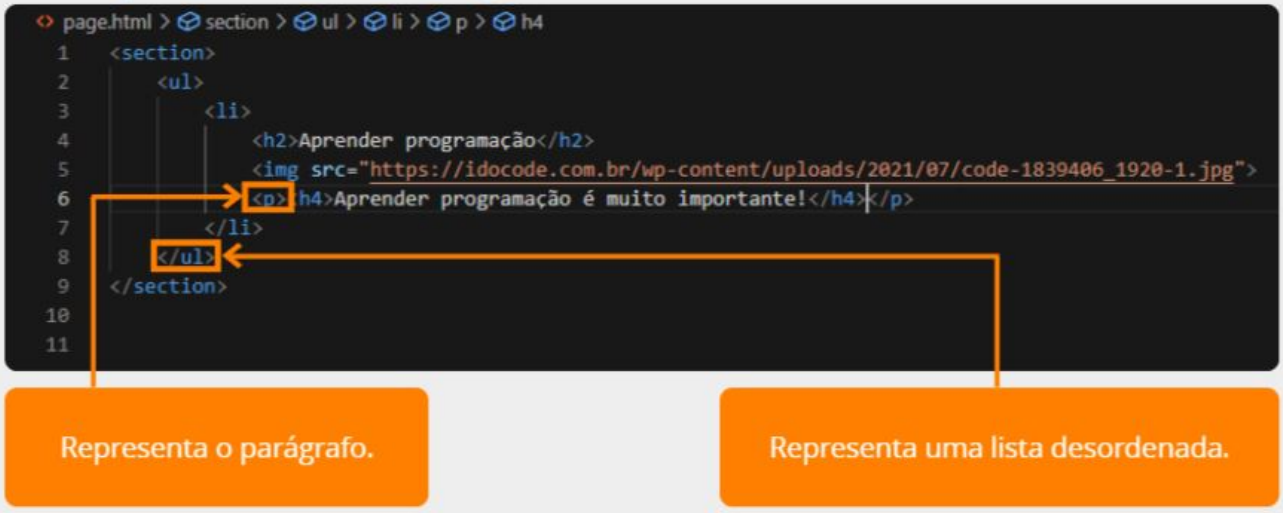
É possível categorizar as tags em dois tipos: o **block-level** (nível de bloco) e o **inline** (em linha). O primeiro ocupa todo o espaço do seu elemento pai e o segundo ocupa apenas o espaço do seu conteúdo, como no exemplo ilustrado.

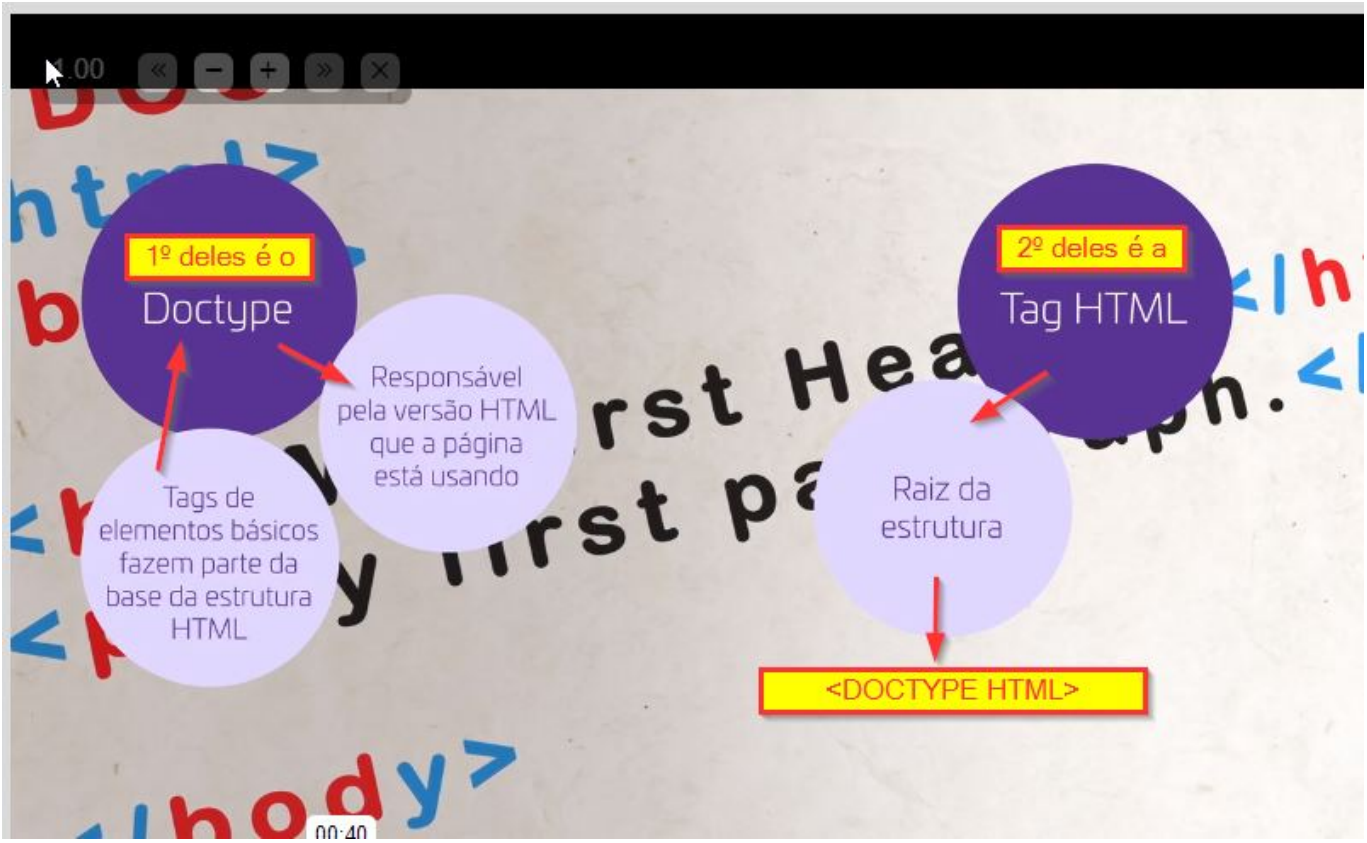


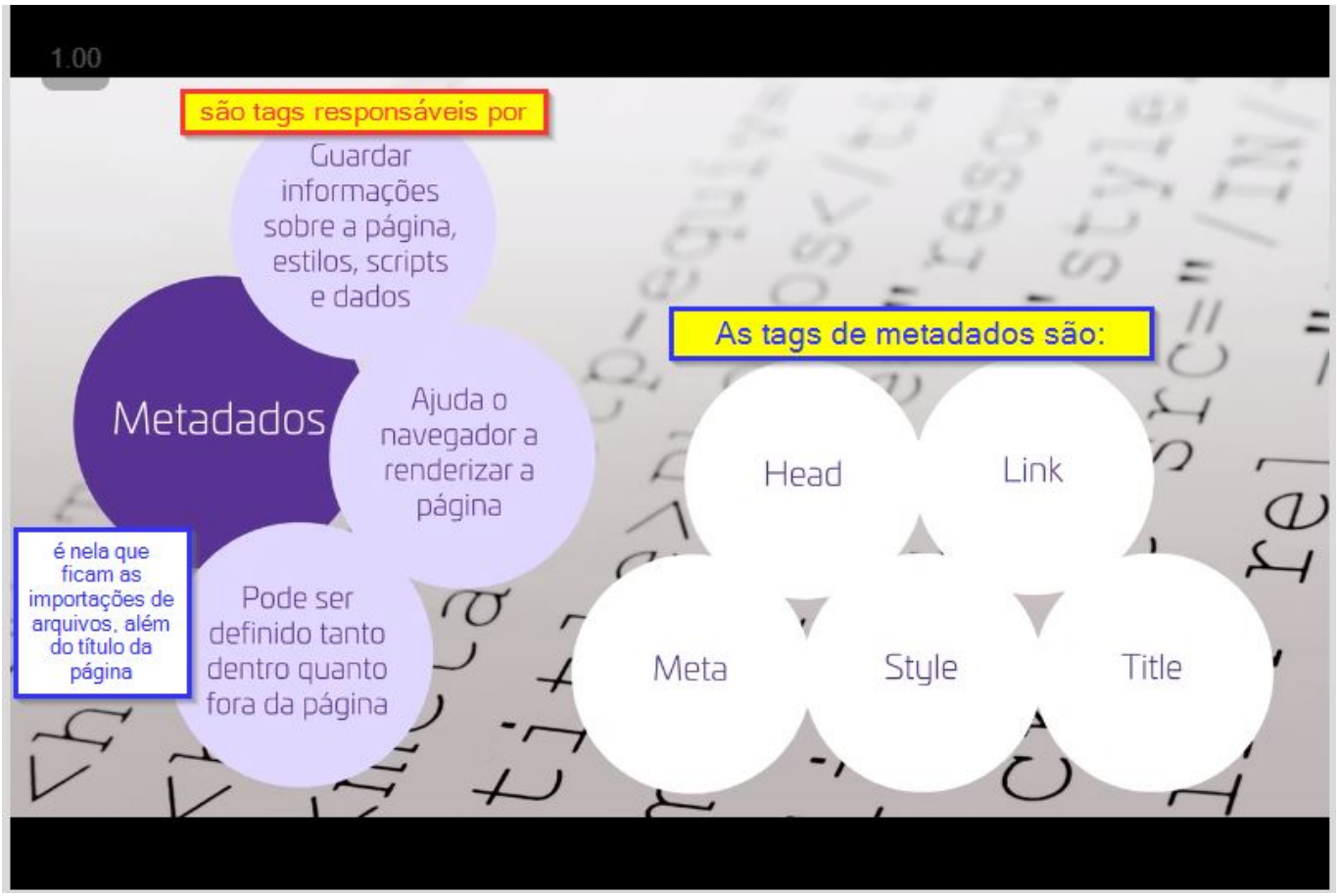
Exemplo de uso das *tags*



Exemplo de uso das *tags*







TAGS de separação de conteúdo

É a tag responsável na organização logica dos conteúdo das página.

TAGS de texto

São responáveis pela impressão do texto para o usuário;

Semântica textual inline

TAGS de multimídia

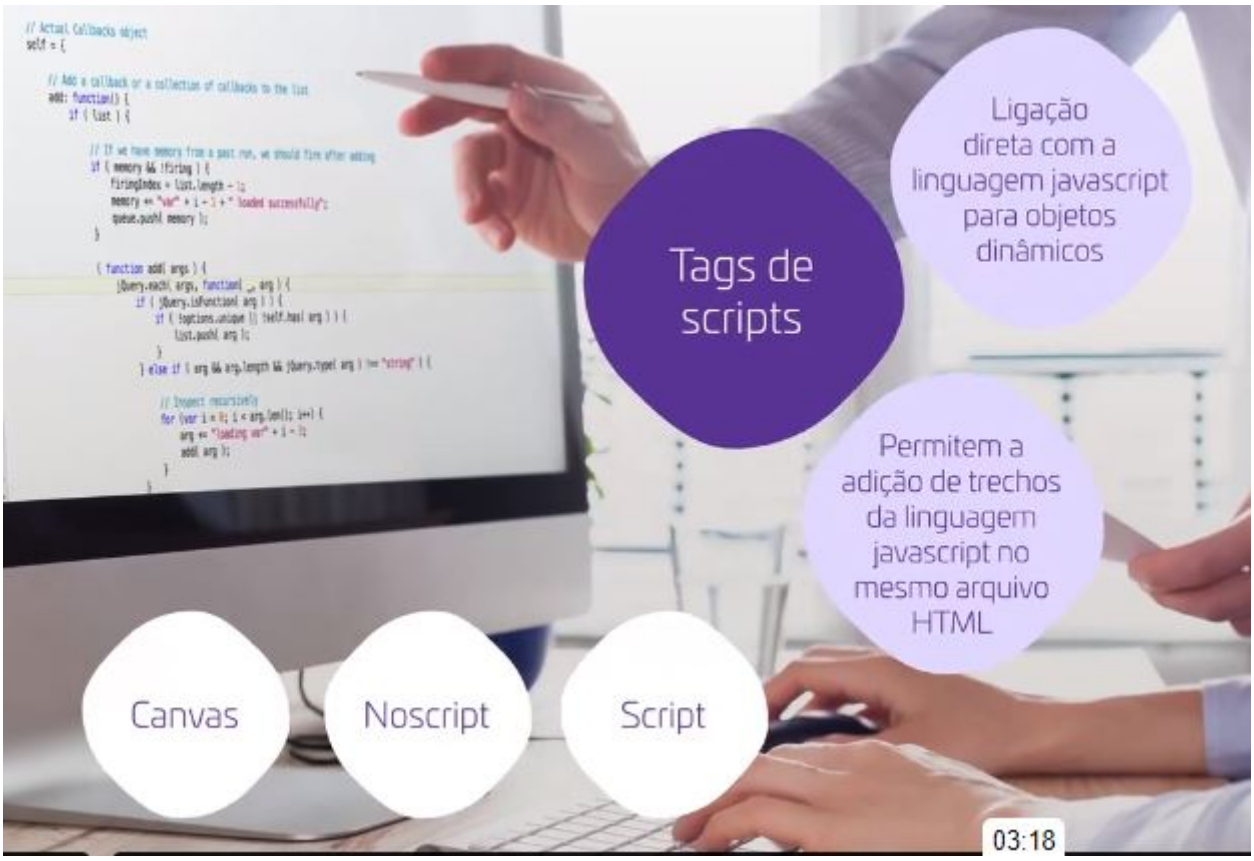
TAGS de tabelas

Eram utilizados nos primeiros sites para estruturação. Atual é usado para criação de estrutura de email.

TAGS de formulário

Responsáveis em compor um formulários.

TAGS de scripts



ALINHAMENTO DOS GRUPOS DECIDIREM SOBRE O LAYOUT

AWS - material extra

SITE1

SITE2

SITE3

SITE4

SITE5 - Usando FIGMA para estruturar layout

SITE6 - Usando FIGMA para estruturar layout 2

COMPREENDER A ESTRUTURA BASE QUE COMPÕE O HTML

Criando a estrutura base da HTML

A estrutura HTML é obrigatória e impressendível para todas as páginas.

Tags necessárias

Para a criação dessa estrutura, são necessárias as tags:

```
<!DOCTYPE html>, <html>, <head>, <meta>, <title> e <body>.
```

As tags possuem hierarquias para que possam ser exibidas e interpretadas pelo navegado.

1ª linha, tag **DOCTYPE**, o qual informa a versão do HTML que será executada; 2ª linha, tag **html**; 3ª dentro da tag **html**, estão as tags **head**(com ou sem script) e a **body**;

***Tag* body**

Recebe toda a informação que será apresentada no site, e para o usuário. **Tudo o que for diferente de metadados** ficará dentro dela.

Leitura Complementar

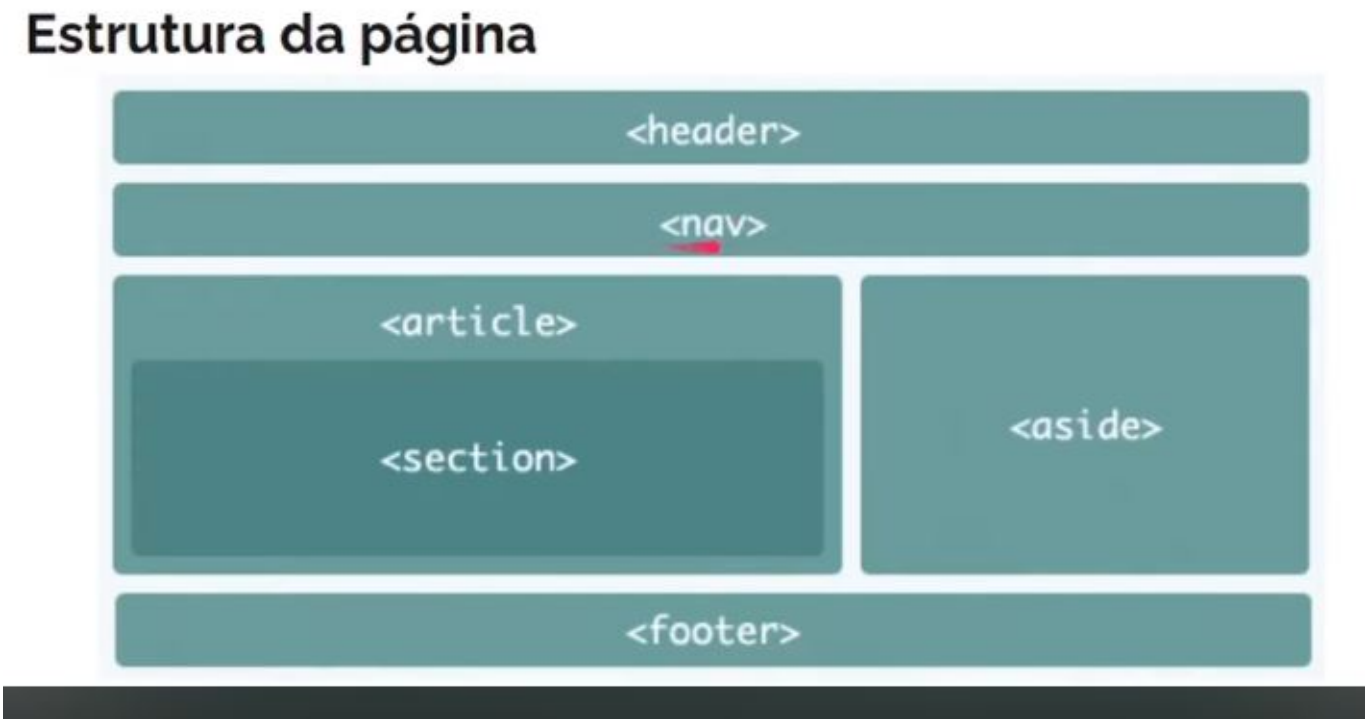
Outras estruturas

Existem outras estruturas para criação de elementos, mas geralmente elas não possuem a formatação apresentada, POR JÁ ESTAREM IMPLICITAS no framework;

HTML SEMÂNTICO

É a forma de sar significado real as tag's. Ao olhar uma tag no corpo do site, vc já sabe o que ele faz. Por exemplo: footer, header e form. Já se sabe o que terá só de ler. Um exemplo de elemento **NÃO SEMÂNTICO**, são **div, span**.

Utilizar a semântica ajuda ao SEO, google ranqueia melhor a página. Ajuda os leitores de tela(deficientes a navegar). Deixa o código mais limpo e de fácil manutenção.

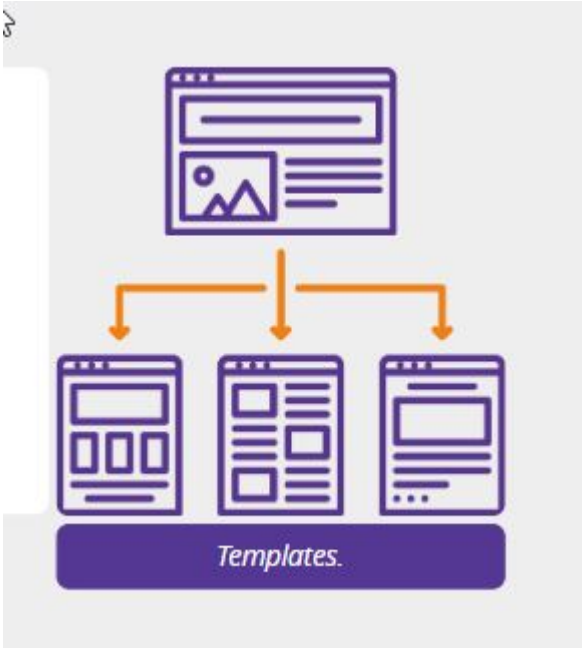


- **header**, fica dentro da tag body, representao cabeçalho. recebe geralmente logo e link;
- **nav**, presenta os links de navegação(menus). Pode estar contido no header;
- **aside**, é tipo uma caixa de conteúdo. independente, pode ou ter ter relação.
- **article**, representa um conteúdo independente dentro do documento. Ex: um artigo de uma páginaa, ou uma pergunta no stackoverflow, por exemplo
- **section**, A tag section é utilizada para marcar as seções de conteúdo de uma página. Com Esse elemento agrupamos de forma lógica nosso conteúdo, separando a informação em áreas diferentes.

ENTENDER COMO CRIAR O CORPO DE UMA PÁGINA

Introdução sobre como criar o corpo de uma página

Além da estrutura da HTML, existem templates, que são modelos. Esses modelos tem conteúdos apresentados de diferentes formas.



Criação do título

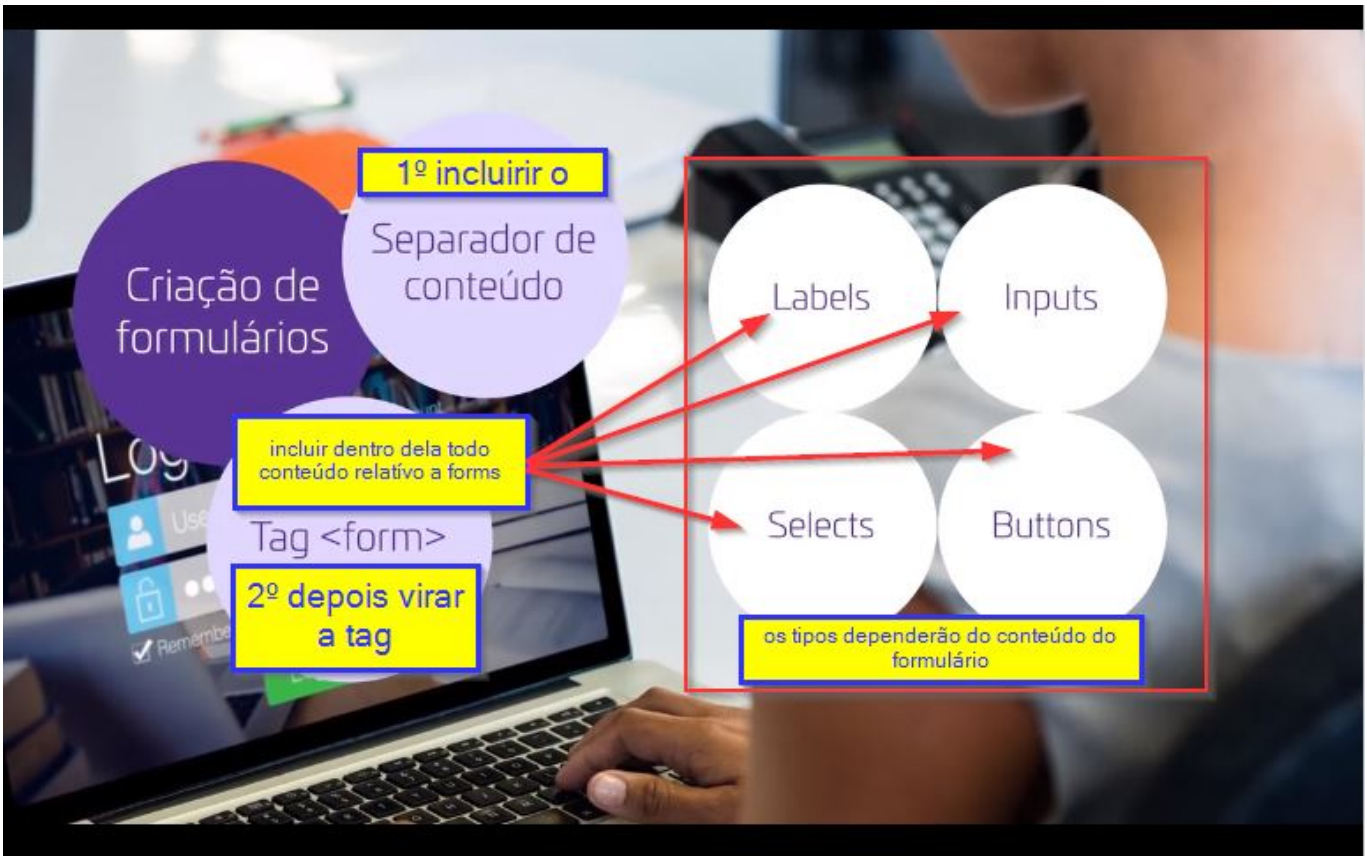
```
<header>
    <h1>Meu Blog</h1>
</header>on>
```

Criação dos textos

Podemos usar diversas formas, SE FOREM SÓ TEXTOS podemos usar **div**, ou a **section**, que representará um bloco de texto.

```
<section>
    <img src="./paisagem.jpg" />
    <h3> Sejam bem-vindo(as)</h3>
    <p> Esse é o meu primeiro blog, portanto gostaria de cocumetar meus
pensamentos e alguns poemas com vocês, epsero9 que gostem.</p>
</section>
```

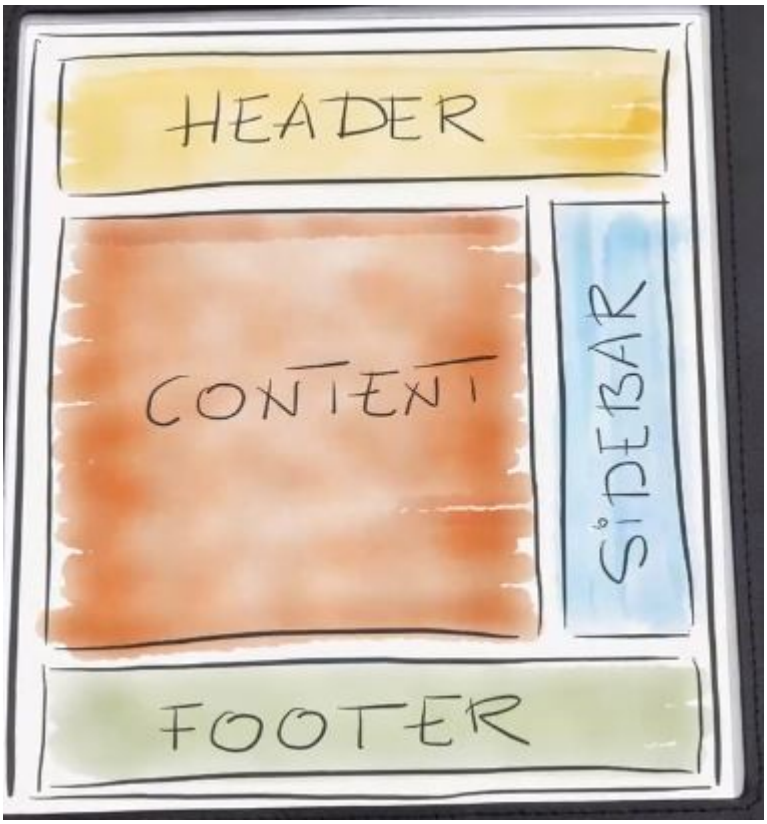
Criação de formulários



Criação da sidebar

Deve ser feita em paralelo com o conteúdo principal.Assim como no texto, podemos usar a **div**, ou outra tag, nesse caso, usaremos a **aside**.

```
<aside>
    <ul>
        <li>Poemas</li>
        <li>Um pouco sobre mim</li>
        <li>Minhas aventuras</li>
    </ul>
<aside>
```



Criação de Tabelas

No início tudo era feito na tabela, pois o CSS ainda estava no ínicio e não suportava; As tabelas são hoje utilizadas na confecção de estrutura de email, o qual contém HTML e TABELAS

slot

Utilizado quando se quer dar dinamicidade a algum elemento/espaço.

Criação do rodapé

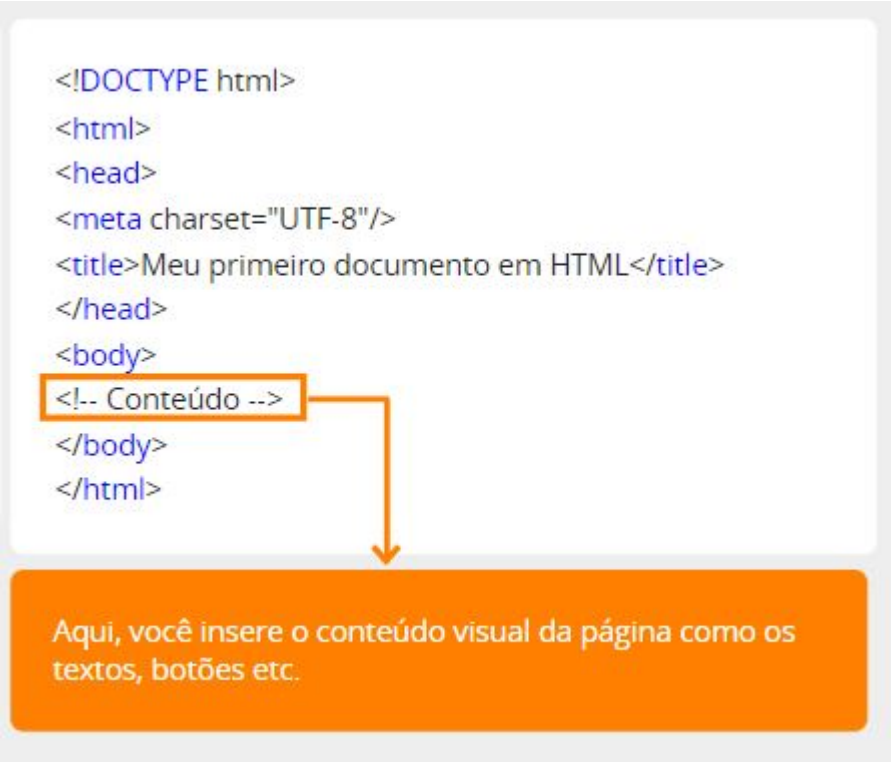
Além desses elementos, podemos incluir um rodapé no blog. Esse é usado para incluir informações sobre contato, links externo e para voltar ao topo da página tag **footer**

```
<footer>
    <span>Contato</span>
    <ul>
        <li><a href=”facebook.com/meublog”>Facebook</a></li>
        <li><a href=”instagram.com/meublog”>Instagram</a></li>
        <li><a href=”twitter.com/meublog”>Twitter></a></li>
 </ul>
</footer>
```

Conclusão sobre criar o corpo da página

Essas são algumas tags que podemos utilizar para o desenvolvimento do nosso documento. Lembre-se de iniciar indicando qual é o tipo de linguagem que será utilizada, que no nosso caso foi a HTML. Para isso, utilizamos a tag

e todo o conteúdo dentro da tag body



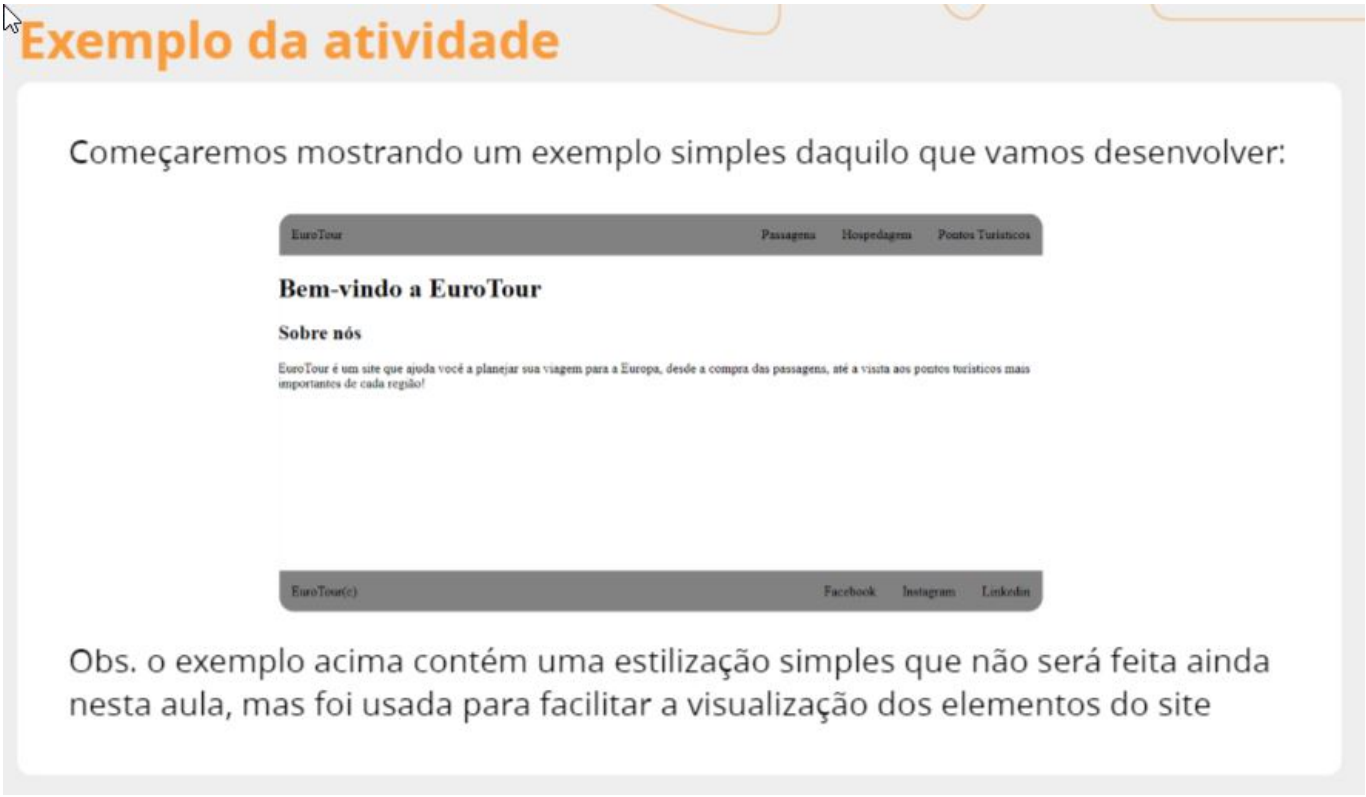
Com a evolução da liguagem de marcação, algumas tags ficaram **obsoletas** por exemplo:

```
* <CONTENT>
* <IMAGE>
* <CENTER>
* <SPACE>
* <DIR>
* <FRAME>
* <APPLET>
* <BLINK>
```

Leitura complementar

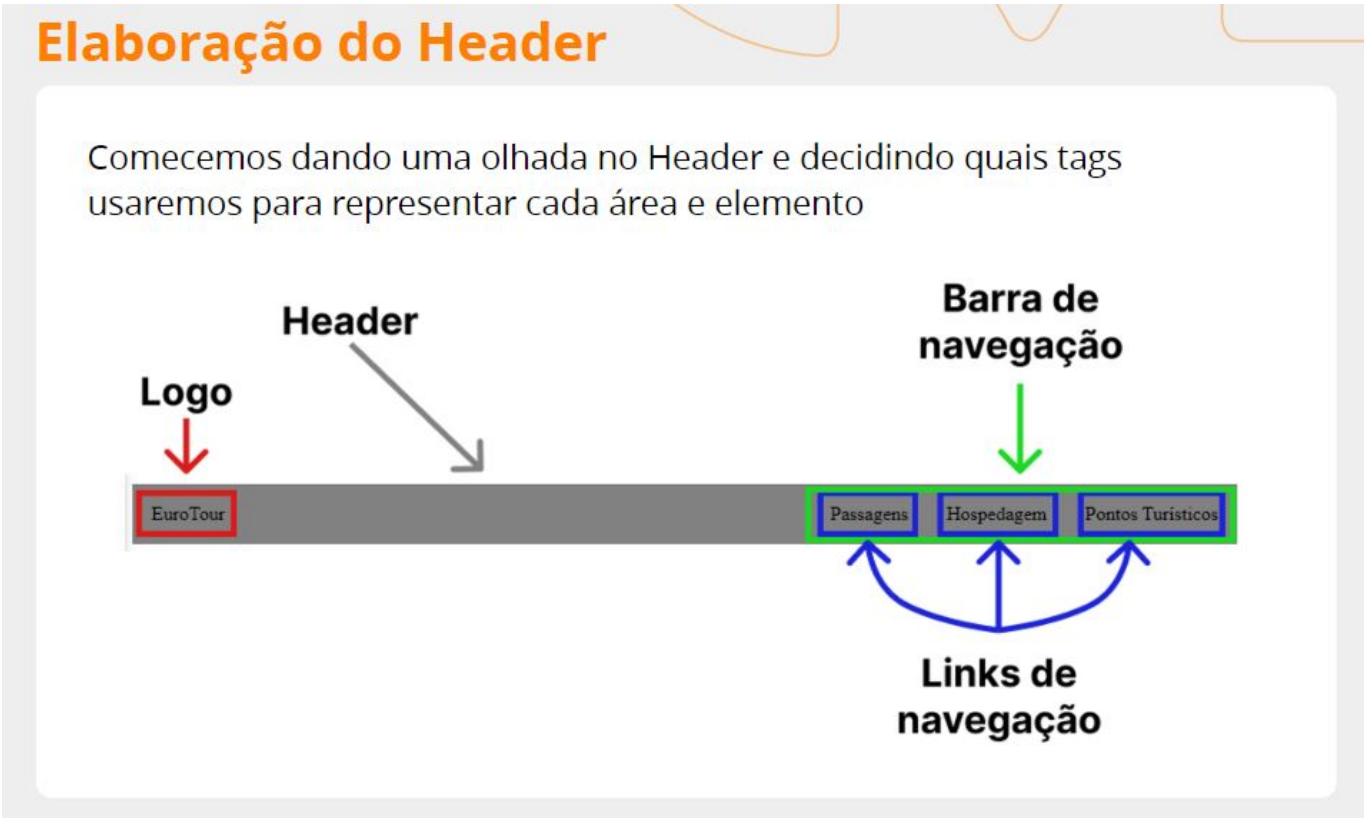
Material complementar AWS - Criando página html

ao criar o arquivo index.html no VSCode nos oferece um atalho para criar a estrutura base de qualquer arquivo HTML. Basta digitarmos um símbolo de exclamação '!' na primeira linha do arquivo, e o programa automaticamente mostrará um pequeno pop-up com duas alternativas: um símbolo de exclamação (!) ou três símbolos de exclamação (!!!). Assim que aparecer o popup, basta apertar a tecla Enter, a tecla Tab, ou clicar diretamente na opção com um símbolo de exclamação (!) do popup. Ao fazer isso, o VSCode criará automaticamente a estrutura base com as tags !DOCTYPE, html, head e body.



estamos focando por enquanto nas **três partes principais de um site**, o **header** (parte cinza no topo), o **main** (título, subtítulo e texto no meio da página), e o **footer** (parte cinza no rodapé da página.

> lembrar que que todo o conteúdo que será de fato exibido ao usuário no site deve estar "dentro da tag body"



Embora sabemos que é possível usar a tag div para todas as seções, é uma boa prática usar as tags respectivas para cada seção:

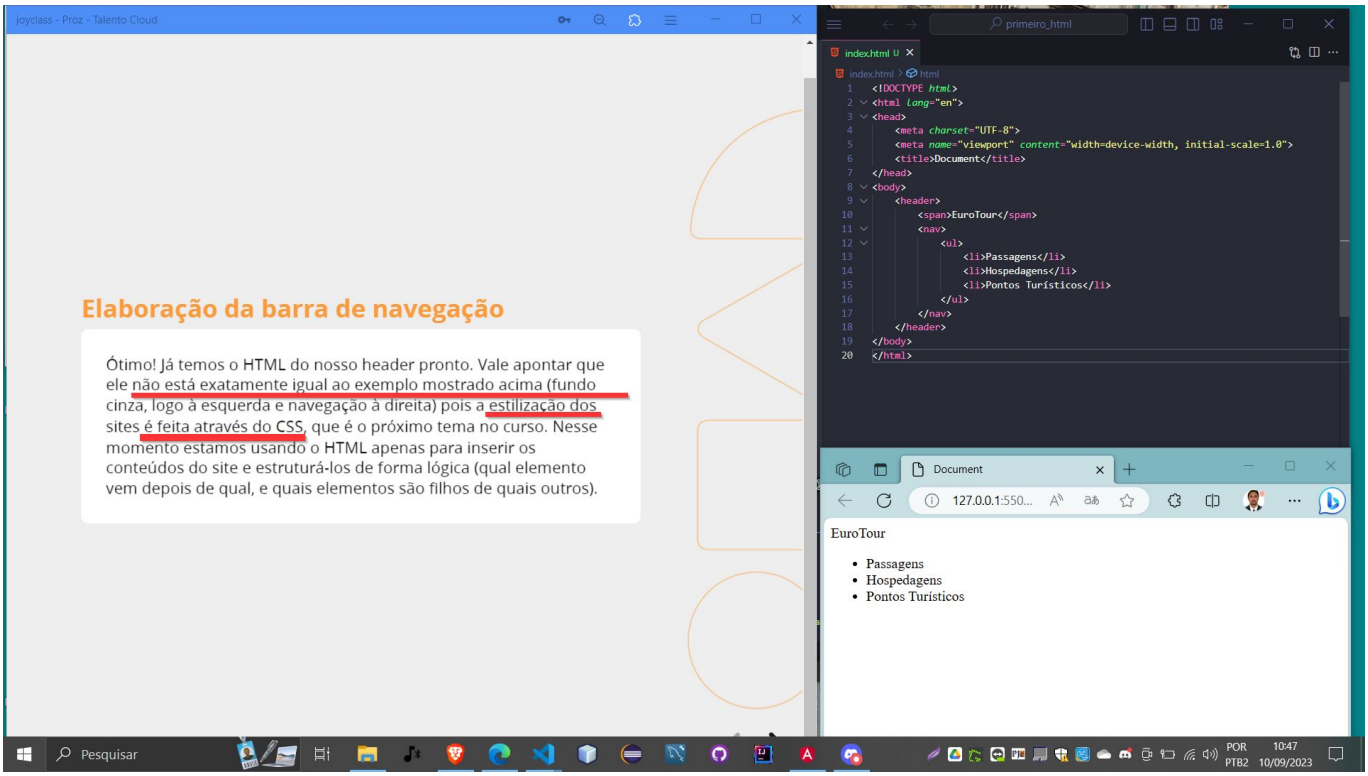
Header - Usaremos a tag <header>, esse header não é mostrado ao usuário.

Logo - Geralmente o logo é uma imagem, porém, nesse exemplo usaremos a tag

Barra de navegação - Usaremos a tag <nav>

Links de navegação - Neste caso, como temos uma lista de elementos sem ordem específica a serem exibidos, usaremos a tag <ul> para criar uma "lista não ordenada", e uma tag <li> para cada item da lista (neste caso, três itens: Passagens, Hospedagem e Pontos turísticos)

Como a tag header terá outros elementos dentro dele (logotipo, navegação, etc.), dizemos que o header é o elemento pai, e todas as tags dentro dele são os elementos filho. Uma boa prática quando começamos aninhar elementos filhos, é deixá-los em linhas diferentes das tags de abertura e fechamento de elemento pai. Usemos a tecla Enter para inserir uma nova linha de código entre as tags <header> e </header>, e dentro delas digitemos a tag span com a palavra "EuroTour" que servirá como logotipo do nosso site:



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>EuroTour</title>
</head>
<body>
  <header>
    <img src="./public/eurotour-logo.png" alt="Logo da EuroTour">
    <nav>
      <ul>
        <li>Passagens</li>
        <li>Hospedagem</li>
        <li>Pontos Turísticos</li>
      </ul>
    </nav>
  </header>
  <main>
    <h1>Bem-vindo a EuroTour</h1>
    <section>
      <h2>Sobre nós</h2>
      <p>EuroTour é um site que ajuda você a planejar sua viagem para a
Europa, desde a compra das passagens, até a visita aos pontos
turísticos mais importantes de cada região!</p>


    </section>
  </main>
  <footer>
    <img src="./public/eurotour-logo.png" alt="Logo da EuroTour">
    <ul>
      <li>Facebook</li>
      <li>Instagram</li>
      <li>Linkedin</li>
    </ul>
  </footer>
</body>
</html>
```

No exemplo apresentado, separamos o conteúdo da página em três partes principais: cabeçalho, conteúdo principal e rodapé. Para isso, dentro da tag body incluímos as tags header, main e footer respectivamente. No header usamos uma tag img para incluir o logotipo da nossa empresa fictícia, e a tag nag para a barra de navegação do site. Na barra de navegação contamos com uma lista não ordenada de elementos que virão ser os links para as outras páginas do nosso site.

O conteúdo principal (tag main) conta com um título (tag h1) e uma seção logo embaixo (tag section). A seção, por sua vez, conta com um subtítulo (tag h2) e um parágrafo (tag p). Finalmente, no rodapé (tag

footer) usamos novamente a tag img para inserir a imagem do logotipo do site, e as tags ul e li para definir os links das redes sociais da nossa empresa fictícia

3.1.0 - APRENDER O QUE É CSS E SUA HISTÓRIA

3.1.1 - O que é CSS e sua história

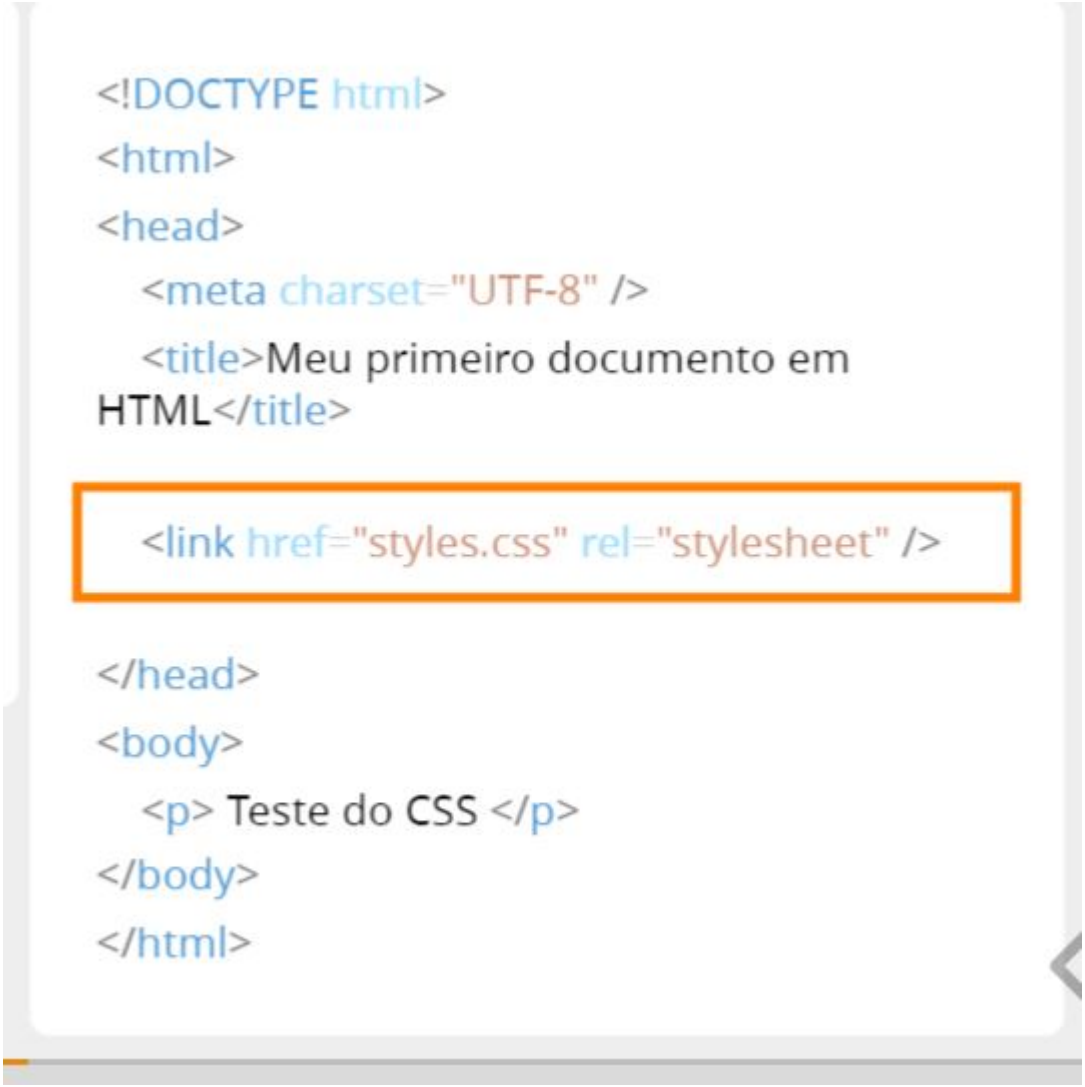
A Cascading Style Sheets (**CSS**), no português **Folhas de Estilo em Cascata** é uma **linguagem de estilização** utilizada em conjunto com a HTML para aplicações web.

Com a evolução do HTML cada navegador fazia da sua forma e acabava por deixar a linguagem mais complexa e dificil de manter. Cada navegador carregava de uma forma, dando instabilidade ao site.

Em 1994, o companheiro de Tim Bernes, Hakon Wium, decidiu criar uma forma mais simples de stilização. A World Wide Web Consortium (W3C) aprovou logo no outro ano e criou um grupo de estudo. 1996 a primeira versão, 1999 a segunda e 2000 a terceira. Que é a mais atual. Vem recebendo atualizações de novas funções.

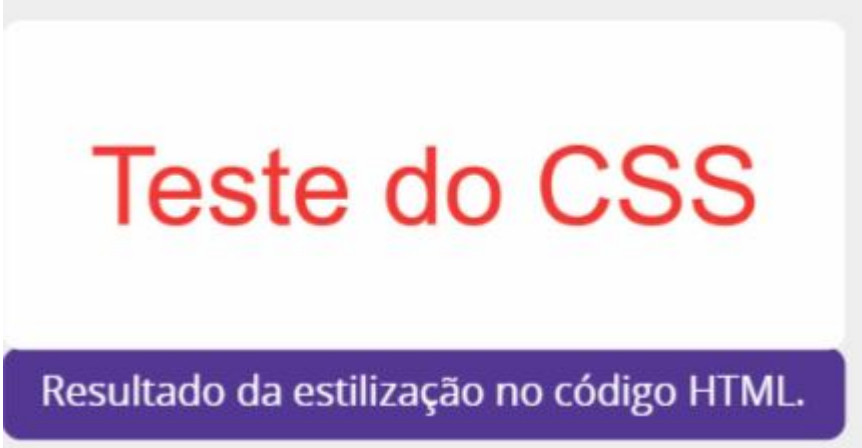
3.1.2 - Como usar a CSS

A CSS **permite aplicar estilos seletivos**. Portanto, vamos aprender como aplicá-los nos elementos da HTML. Após a criação de toda a estrutura em HTML da página, vamos **criar um arquivo chamado styles.css**. Nele, colocaremos todas as configurações de estilo. Na tag head da nossa página, vamos fazer a importação do arquivo de estilos



Dentro do arquivo .css incluir quais elementos receberão a stilização

```
p {
    font-family: sans-serif;
    color: red;
    font-size: 50px;
}
```



> ATENÇÃO! Vale ressaltar que, em alguns momentos, **será possível a aplicação de estilos através de uma hierarquia**. Por exemplo, em uma section, há uma div na qual há um elemento p para estilizar. Para isso, usamos a linha de código section div p {} e colocamos os estilos desejados dentro das chaves.

```
<body>
    <section>
        <div><p> Teste do CSS </p></div>
       </section>
</body>
```

```
section div p {
    font-family: sans-serif;
    color: blue;
    font-size: 50px;
}
```

Por exemplo, em uma **section**, há uma **div** na qual há um elemento **p** para estilizar. Para isso, usamos a linha de código **section div p {}** e colocamos os estilos desejados dentro das chaves.

> ATENÇÃO!

Para aplicar um mesmo estilo para mais de um elemento ou classe, devemos separá-los por vírgulas, ou seja: p, span, a {}. Veja o exemplo

```
p, span, a {
    font-family: sans-serif;
    color: blue;
    font-size: 50px;
}
```

> ANOTAÇÃO: Não há necessidade de uso de vírgulas. Conforme exercício.

3.2 - CONHECER CLASSE E ID E QUANDO UTILIZAR

3.2.1 - Definições e utilidades

Para fazer a estilização na CSS, **precisamos utilizar seletores**. Apesar das atualizações e das especificações da CSS, as formas de seleção para a aplicação de estilos continuam as mesmas. Elas são:

- através de uma **class**;

- através de uma **id**;
- através das **tags**;
- através do **atributo** de um elemento.

Class, Utilizar class é **muito mais complicado**, pois **uma class é utilizada quando é necessário aplicar um estilo para mais de um elemento**. Ela também serve para a aplicação de estilos mais genéricos, como uma cor ou tamanho de fonte. A class é muito presente nos frameworks para componentes.

> Lembrando que um elemento pode ter mais de uma class.

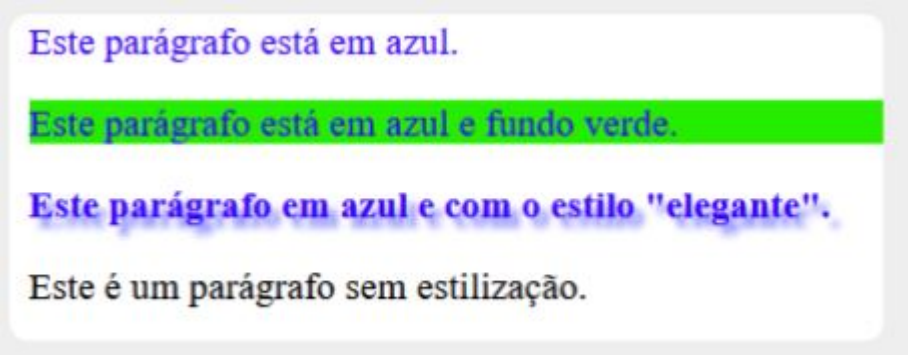
Para aplicar uma class, no arquivo .css , **é necessário colocar um `ponto` na frente do nome da class que será carregado no html** Esse nome será um apelido, um alias, que será dado para ela. Veja o exemplo:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8"/>
<link rel="stylesheet" type="text/css" href="styles.css" media="screen" />
<title>Exemplo do uso do seletor: class</title>
</head>
<body>
    <p class="azul">Este parágrafo está em azul.</p>
<p class="azul verde-bg">Este parágrafo está em azul e fundo verde.</p>
<p class="azul elegante">Este parágrafo em azul e com o estilo "elegante".</p>
<p>Este é um parágrafo sem estilização.</p>
</body>
</html>
```

```
.azul {
  color: rgb(55, 0, 252);
}

.verde-bg {
  background: rgb(43, 235, 4);
}

.elegante {
  font-weight: bold;
  text-shadow: 4px 4px 3px #77f;
}
```



id, também é muito presente nessa aplicação. A diferença é que ela só será utilizada quando apenas um estilo for necessário.

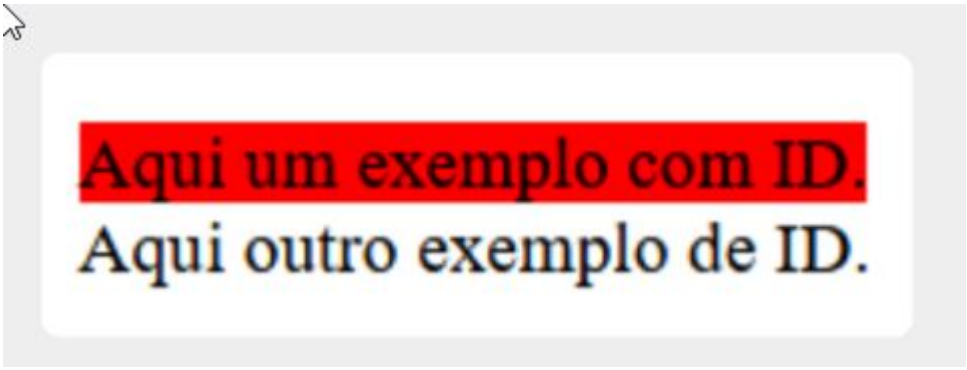
> porque um elemento não pode ter mais de uma id.

Então, **quando não houver repetições, deve-se utilizar a id**.

Para aplicar uma id, é necessário colocar uma hashtag na frente do nome que será dado para ela. Veja o exemplo:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8" />
    <link rel="stylesheet" type="text/css" href="styles.css" media="screen" />
    <title>Exemplo do uso do seletor: ID</title>
</head>
<body>
    <br>
    <span id="exemploId">Aqui um exemplo com ID.</span>
    </br>
    <span>Aqui outro exemplo de ID.</span>
</body>
</html>
```

```
span#exemploId {
  background-color: red;
}
```



Seleção de atributo quando percebe-se que não há a necessidade de usar uma id ou class e se o elemento tiver um atributo, a seleção pode ser feita através desse. Assim, se combina elementos baseados no valor de um atributo dado.

Para aplicar a seleção através de um atributo, chamamos o elemento e entre [] colocamos o atributo que queremos alterar.

```
a[title] {

  color: red;

}
```

No exemplo a seguir, é possível adicionar uma cor ao título de um link através de um atributo.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8" />
    <link rel="stylesheet" type="text/css" href="styles.css" media="screen" />
    <title>Exemplo do uso do seletor: atributo</title>
</head>
<body>
    <ul>
        <li><a href="#internal">Internal link</a></li>
        <li><a href="http://example.com">Example link</a></li>
        <li><a href="#InSensitive">Insensitive internal link</a></li>
        <li><a href="http://example.org">Example org link</a></li>
      </ul>
</body>
</html>
```

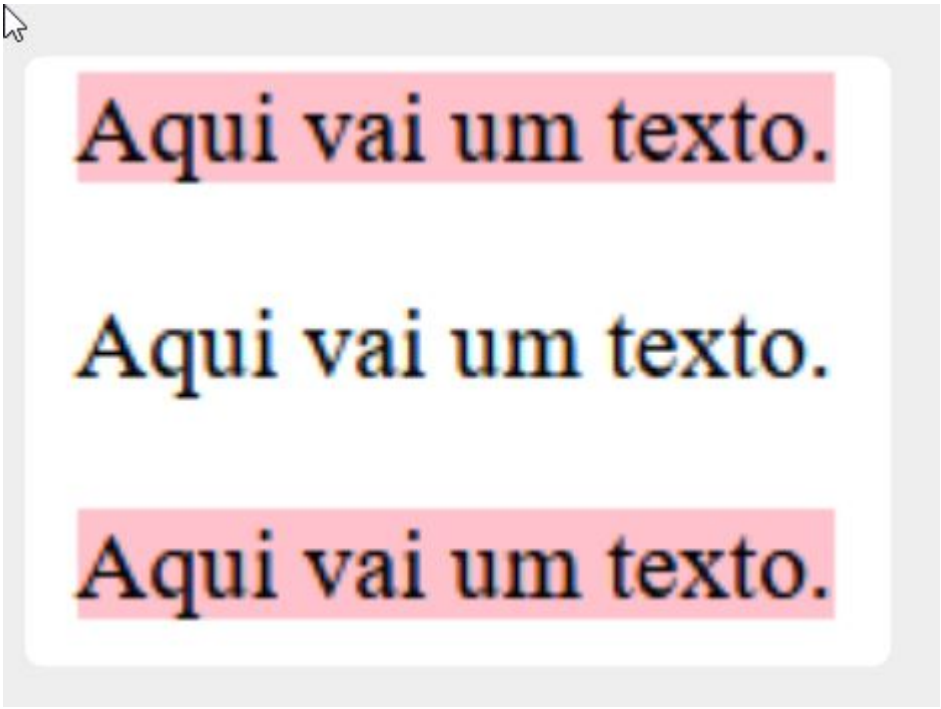
```
a {
  color: blue;
}
/* Links internos, começando com "#" */
a[href^="#"] {
  background-color: gold;
}
/* Links com "example" em qualquer lugar da URL */
a[href*="example"] {
  background-color: silver;
}
/* Links com "insensitive" em qualquer lugar da URL,
   independentemente da capitalização */
a[href*="insensitive" i] {
  color: cyan;
}
/* Links com final ".org" */
a[href$=".org"] {
  color: red;
}
```

Tags, Este seletor, utilizamos as tags que declaramos no corpo do html, como no exemplo a seguir.

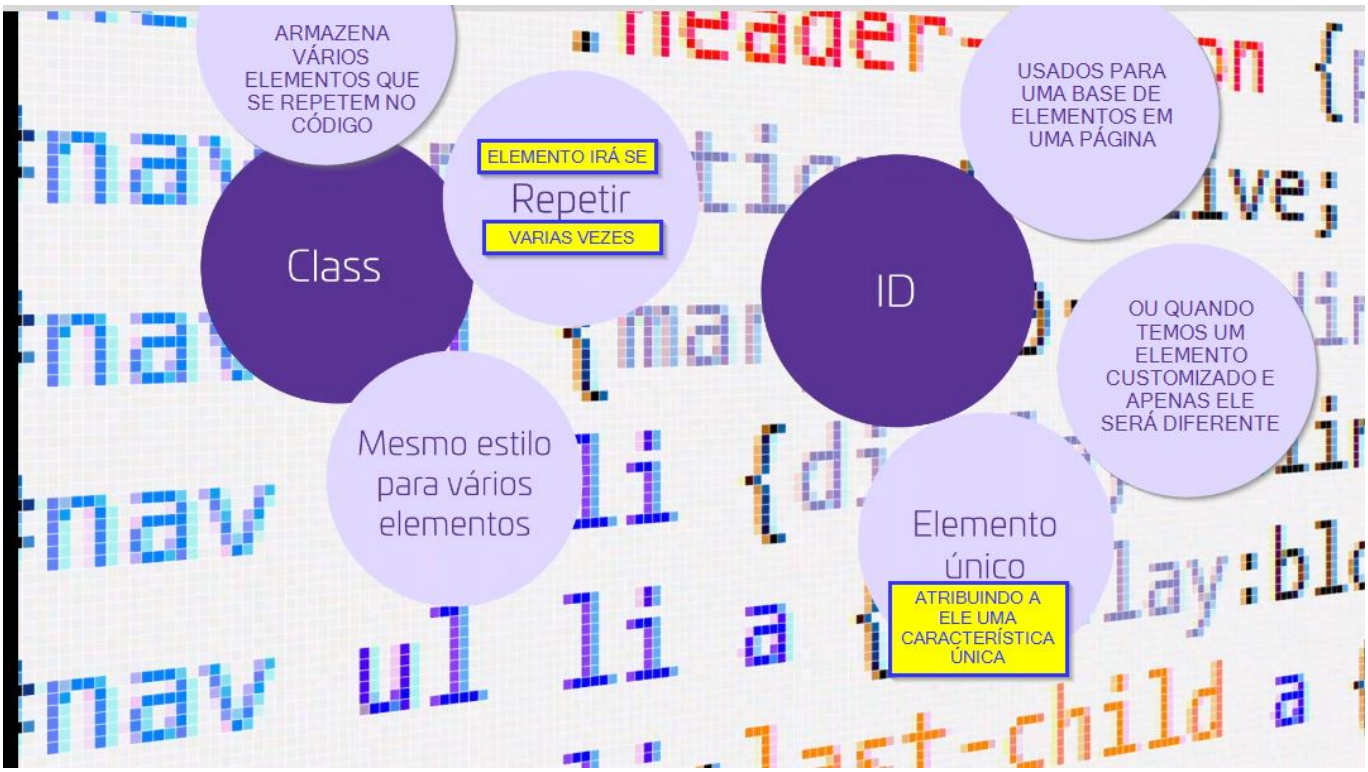
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8" />
    <link rel="stylesheet" type="text/css" href="styles.css" media="screen" />
    <title>Exemplo do uso do seletor: atributo</title>
</head>
<body>
    <span>Aqui vai um texto.</span>
    <p>Aqui vai um texto.</p>
```

```
    <span>Aqui vai um texto.</span>
</body>
</html>
```

```
span {
  background-color: pink;
}
```



> DIFERENÇA ENTRE CLASS E ID



Para esses elementos que se repetem durante o código. Diversas chamadas de um mesmo componente com as mesmas caracteristicas de estilo. Podemos armazenar esses elementos dentro de uma classe e assoliala ao componente.

Muitas fezes é usando pela falta de tipos de tags para separação de conteúdo é utilizada as tagas Class e ID para realizar essa separação.

> ATENÇÃO! Na criação de nomes de Class e ID, devemos seguir a convenção de escrever tudo minúscula. Fazendo uso do **kebab case**. Todas as letras são minusculas e separação é feita por traço/hífem

Onde haver ID poderá haver Class, um ELEMENTO PODE TER MAIS DE UMA CLASS, e o ID é único.

Através do JavaScript poderemos acessar o elemento pelo ID, por uma função. Class também poderá ser acessada.

3.3 - ENTENDER AS PROPRIEDADES BÁSICAS DO CSS

3.3.1 - Propriedades básicas da CSS

3.3.1.1 - Espaçamento

margin: 10px 10px 10px 10px; para **espaçamento externo** ao elemento top, right, bottom, left

padding: 10px 10px 10px 10px; para **espaçamento interno** ao elemento top, right, bottom, left

3.3.1.2 - Texto

font-family: atribui qual fonte será utilizada.

font-size: atribui o tamanho da fonte,

- **pixels**, Medida de valor absoluta, semelhante a centímetros ou polegadas, mas usando como referência os pixels do nosso dispositivo. Basta escrever o número seguido das letras 'px' Exemplo: font-size: 16px;
- **EM**, Unidade de medida relativa ao "elemento pai" Se temos um elemento pai com, por exemplo, um font-size de 48px, e atribuímos ao elemento filho dele um valor de 1em, esse 1em será equivalente a 48px. Se o valor do font-size do elemento pai mudar para 32px, o 1em passará a ter um valor de 32px.
- **REM**, Unidade de medida relativa ao "elemento root", ou o elemento html. Ambos tem como padrão o valor 16px, ou seja, **1rem equivale a 16px**, 0.5rem equivale a 8px, 2rem equivale a 32px, e assim por diante 16px é considerado o valor padrão por ser grande o suficiente para leitura na maioria de ispositivos (smartphones, tablets, notebooks e desktops), porém, é possível alterá-lo nas configurações dos dispositivos ou atribuindo um novo valor com CSS ao font-size do elemento html.

> Isto é útil, por exemplo, para questões de acessibilidade. Se pessoas com baixa visão, que deixam o tamanho padrão da letra maior nos seus dispositivos, acessarem nosso site, o tamanho da fonte nele se ajustaria automaticamente ao padrão dos seus dispositivos ou navegadores (diferente de medidas absolutas como pixels)

font-style: atribui itálico por exemplo. **Por padrão, os elementos HTML "herdam" seus estilos dos elementos pai.** Para remover é só incluir no filho o **font-style: normal;**

color: atribui cor ao texto; aceita:

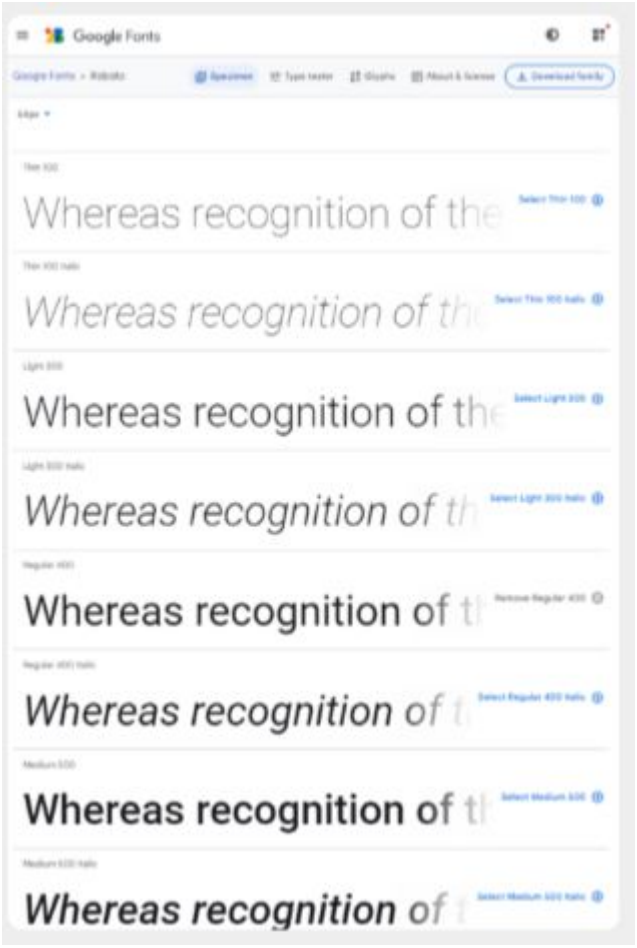
- nome da cor: escrevendo o nome dela "red" "blue";
- RGB (0,0,0); O sistema atribui um valor de 0 a 255 para cada uma dessas "tonalidades base", onde 0 é a ausência total da cor, e 255 a intensidade máxima dela. RGBA, pode-se incluir transparência no último campo (0,0,0,**0**) zero totamente transparente e 1 totalmente opaco.
- Hexadecimal (As cores são representadas por um sistema de numeração em base 16, ou seja, com 16 símbolos do 0 ao F - Dessa forma, conseguimos representar os valores do 0 ao 255 do sistema RGB apenas com dois dígitos (ex. O número 255, em base 10, é representado como FF em base 16))

text-align: podendo ficar center, rigth, etc.

font-weight Podemos aumentar ou diminuir o "peso" ou "intensidade" de qualquer texto usando a propriedade CSS font-weight. A propriedade font-weight aceita valores predefinidos (ex. normal, bold, lighter) ou valores numéricos entre 100 e 1000, com acréscimos de 100 (ex. 100, 200, 300...).

> É importante lembrar também que, se estivermos importando fontes do Google Fonts, podemos escolher também quais pesos e estilos queremos importar.

Os filhos herdam o valor dado. Para resetar utiliza a configuração "normal"



3.3.1.3 - Abrituir fundo

background: url que buscaraá a imagem ou uma cor será definida.

3.3.1.4 - Hiperlink

Ao inserir fica sublinhado, para tirar o sublinhado deve-se utilizar **text-decoration: none**

3.3.1.5 - Display

Definirá o tipo de renderização que será utilizada no elemento, como esse elemento ficará na tela. **display:block;** Existem várias propriedades: absolute, relative, block, flex, none; **block é default**

> Atenção Desativando, todos os elementos descendentes irão sumir.

3.3.1.6 - FlexBox

Muito utilizada em organizar elementos em forma RESPONSIVA. Muito utilizada principalemente por frameworks.

utiliza-lá dando **display:flex**

Propriedades utilizadas junto do display:flex são :

- **justify-content**; alinha horizontalmente.
- **align-items**; alinha verticalmente.

- **flex-direction**; que direção devem seguir.
- **flex-wrap**; se os elementos devem ou não quebrar e para onde devem ir se o fizerem.

3.3.2 - AWS - CSS Propriedades Básicas

[Como iniciar CSS - Flex Box]https://youtu.be/gOMK_xruAqc Muito bom

[Aprenda FlexBox com FlexFrog]https://youtu.be/7uGDoJN0tAA

> link para id: Nomeia a div com id e faz um '<a href=#"colocando aqui o nome do id>One' ao clicar faz o pulo pra div. uso no css o "scroll-behavior:smooth" para fazer uma transição suave

3.3.3 - Unidades de medida

determinam o tamanho de qualquer elemento;

absolute = será usado só **pixel**, não se baseia em nada. imutável por nada até intervenção.

relativas

- **em**, o elemento se comporta conforme o elemento Pai. Se o pai tiver font-size:50px e o **p** do filho tiver font-size: 1em. o parágrafo do filho terá o tamanho igual ao pai. Se o filho tiver 0.25em o filho terá 25% do tamanho do pai. 2em seria o dobro do pai, no caso 100px
- **rem**, deixa de se basear no elemento mais próximo e passa a ser baseado no root do documento. Bom uso para experiencia do usuário.
- **vw**, (**v**iew**w**idth) 1vw é 1% da largura do view port.
- **vh**, (**v**iew**h**eight), mesmo padrão. do vw
- **%**, baseada no elemento pai, pai com 200px, e o filho com 50%. filho terá
- **vmin**, é relativo ao menor parte do viewport 2vmin, 2% da menor parte
- **vmax**, maio parte.

> navegador por padrão deixa 16px de tamanho de fonte. 100% seria 16px

3.4 -IMPORTAÇÃO E MANIPULAÇÃO DE FONTES

3.4.1 - CSS e fontes tipográficas

usaremos o Google Fonts para importar fontes nos nossos projetos. Google Fonts é um serviço gratuito com quase 1000 tipos de fontes diferentes. Ele não requer nenhum tipo de cadastro, login, ou instalação, e é amplamente usado pela comunidade de desenvolvimento web.

3.5 - CSS

3.5.1 - HandsOn Luis

Projeto zipado - NÃO ACHEI.

Espaçamento padrão, entré título e subtitulo. Há um espaçamento do proprio navegador, tipo uma margem do body padrão.

São estilos padrões e que podem ser removidas. Utilizar para toda a página usando dentro do seletor *{} afeta **todos** os elementos. Alguns elementos vem com padding padrão.

```
margim:0;
padding:0;
```

> ATENÇÃO!! O DISPLAY: FLEX VAI NO ELEMENTO PAI!!! PARA AFETAR OS FILHOS. Por exemplo se quer colocar os li um ao lado do outro, deve-se atribuir diplay:flex ao pai.
>
> Para o header e os menu fiquem um ao lado do outro deve aplicar o display:flex no pai de ambros, que seria o header no exemplo;

ul quando pede para centralizar o li não ocupa todo o espaço do Nav, deve-se dizer na propriedade que o ul vai ocupar toda a altura do pai Nav. Aplicar no ul o **height:100%** da altura disponível do pai dele.

3.6 - REPPONSIVIDADE