



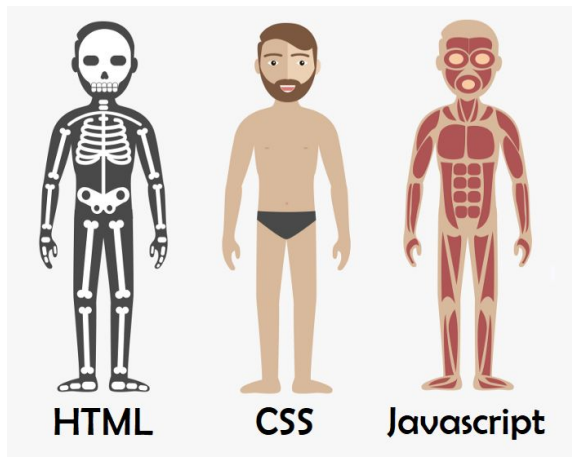
otterwise



# HTML, CSS e JS

# HTML, CSS e JS

- HTML: é como o esqueleto de um site. Ele dita apenas como o site é estruturado;
- CSS: como se fosse a pele do site. O css que deixa o seu com um aspecto mais bonito e dando vida a ele.
- JS: são os tendões e músculos do site. O javascript é quem comanda como as interações vão acontecer e o comportamento que o site tem que ter de acordo com cada interação do usuário.



# O QUE É HTML?

# O que é HTML?

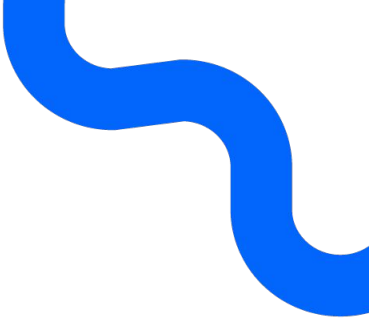


- HTML: Abreviação de **HyperText Markup Language** (Linguagem de Marcação de Hipertexto).
- Linguagem de marcação utilizada na construção de **sistemas e páginas web**.
- Os documentos do tipo HTML são **interpretados pelo navegador** para exibir a página.



# ESTRUTURA BÁSICA

# Estrutura Básica



```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="UTF-8" />
5      <title>Título</title>
6    </head>
7    <body>
8      <!-- Conteúdo -->
9      <h1>Teste</h1>
10   </body>
11 </html>
12
```

# DOCTYPE



- `<!DOCTYPE html>`
- **Não é uma tag HTML.**
- Informa para o navegador que tipo de documento o arquivo possui.
- Documentos HTML devem começar com essa declaração de tipo.



# Elementos (tags HTML) estruturais



- **<html>**[conteudo]**</html>**
  - Raiz de um documento HTML.
  - Contém todos os outros elementos da página.
- **<head>**[conteudo]**</head>**
  - Espaço para informar os metadados do documento, como **título, scripts, estilos e idioma**.
- **<body>**[conteudo]**</body>**
  - Onde ficará o conteúdo do documento HTML.

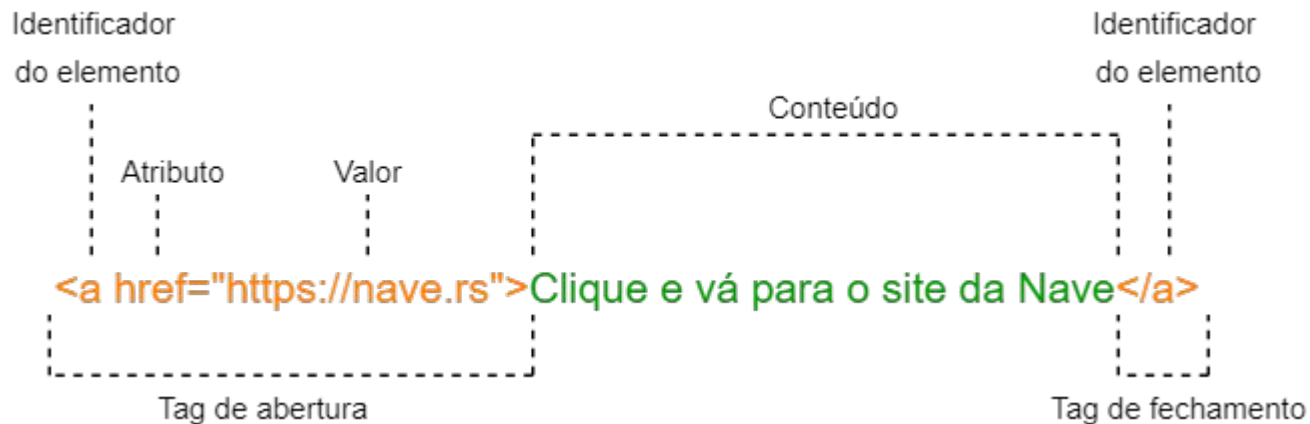


## Elementos (tags HTML) da head

---

- **<meta** [atributo=valor]>
  - Define informações de metadados como a codificação de caracteres usada na página.
- **<title>**[titulo\_da\_pagina]**</title>**
  - Título da página que aparecerá na aba do navegador.
- **<link** [atributo=valor]>
  - Utilizado para carregar um recurso externo como uma folha de estilos.
- **<script>**[script\_JS]**</script>**
  - Utilizado para carregarmos os scripts executáveis ou definirmos um script para ser executado.

# Sintaxe



# Elementos (tags HTML) da body



- **h1 - h6**
  - Elementos de cabeçalho e título de seção. **h1** sendo o mais importante (maior destaque) e **h6** o menos (menor destaque).
  - Exemplo: `<h1>[conteudo | titulo_da_secao]</h1>`
- `<a href=[url] [atributo=valor]>[conteudo | texto_do_link]</a>`
  - Elemento para criar hiperlink. Normalmente utilizado para navegar para outra página.



## Elementos (tags HTML) da body

---

- **<p>**[texto]**</p>**
  - Elemento de parágrafo para escrita de textos na página.
- **<div>**[conteudo]**</div>**
  - Elemento usado para agrupar outros elementos sem influência na estrutura e semântica da página.
- **<img src=[url | path] [atributo=valor]>**
  - Elemento usado para carregar uma imagem na página.

# Atributos básicos das tags HTML



- **id**
  - Identificador único de um elemento HTML no documento.
  - Utilizado como seletor para a estilização desse elemento (CSS).
  - Utilizado como seletor para a manipulação desse elemento (JS).

# Atributos básicos das tags HTML



- **class**
  - Especifica a classe do elemento HTML no documento.
  - Não é obrigatoriamente único.
  - Pode ser usado como seletor para estilização ou manipulação, selecionando todos os elementos de mesma classe.



# NAVEGAÇÃO



# Navegação entre Páginas



- Podemos criar uma aplicação ou site com mais de uma página, onde cada uma delas terá um propósito específico.
- Exemplo:
  - Site com as páginas: Home, Contato, Sobre Nós, etc...
- Para uma boa experiência do usuário é necessário que hajam links entre as páginas para que seja possível navegar pela aplicação.
- Da mesma forma, pode ser necessário criar um link de navegação da nossa página para um site externo.



# ELEMENTOS HTML

# Listas



- `<ul>[items]</ul>`
  - Cria uma lista de itens sem ordem.
- `<ol>[items]</ol>`
  - Cria uma lista de itens ordenada.
- `<li>[conteudo]</li>`
  - Item de uma lista ordenada ou não.

# Elementos Semânticos



- **<header>**[conteúdo]**</header>**
  - Pode ser usado para definir o conteúdo de cabeçalho da página dentro do *body* ou como cabeçalho de uma seção (*section*).
- **<main>**[conteúdo]**</main>**
  - Define o conteúdo principal dentro do *body*. Para um bom uso o seu conteúdo deve ser único na página.
- **<footer>**[conteúdo]**</footer>**
  - Informações de rodapé do site como: copyright, autoria, links relacionados, endereço, contato, etc...

# Elementos Semânticos



- **<aside>**[conteúdo]**</aside>**
  - Utilizado para dar destaque à um conteúdo indiretamente ligado ao conteúdo principal, pode ser usado para uma citação ou um link relacionado.
- **<section>**[conteúdo]**</section>**
  - Elemento usado para separar os elementos da página em grupos de conteúdos similares.
- **<nav>**[conteúdo]**</nav>**
  - Indica uma **seção específica** para links de navegação.



# EXERCÍCIO

## Exercício



1. Crie um **site sobre você**. O site deve conter duas páginas: Home e Skills. Implemente a navegação entre as páginas.
2. Implemente a página Home indicando suas informações, experiências profissionais e pessoais.
3. Desenvolva a página de Skills com uma lista de hard skills e soft skills que você acredita que são boas para mostrar para recrutadores.

**Observação:** Na realização do exercício utilize as tags semânticas que estudamos. Sinta-se à vontade para incrementar as páginas com conteúdos que achar interessante.

# O QUE É CSS?



# O que é CSS?



- **Cascading Style Sheets** - Folhas de Estilo em Cascata.
- Possui uma sintaxe própria.
- Ferramenta utilizada para dar estilo à páginas HTML.
- Descreve como os elementos devem ser mostrados na tela.
- Pode ser utilizado de forma **inline**, **interna** e **externa** no documento HTML.
- Com a ferramenta de inspeção do navegador podemos ver o CSS da página aplicado aos elementos.

# Sintaxe CSS



Seletor	Valor
⋮	⋮
h1	{ color: #424242; }
⋮	
Propriedade	



# PROPRIEDADES

# Propriedades Básicas



- font-size
  - Modifica o tamanho da fonte do elemento.
- font-weight
  - Modifica a finura/grossura da fonte (negrito, normal, etc...)
- line-height
  - Modifica o espaçamento entre as linhas de um elemento.
- color
  - Modifica a cor do elemento.



# Propriedades Básicas

- `margin`
  - Adiciona um espaço externo no elemento.
- `padding`
  - Adiciona um espaço interno no elemento.
- `background-color`
  - Muda a cor do fundo.
- `width`
  - Define a largura do elemento.
- `height`
  - Define a altura do elemento.
- `display`
  - Define como o elemento deve ser mostrado na tela.



# SELETORES

# Seletores



- Forma de selecionar os elementos HTML para aplicação de estilo.
- Além dos seletores de elemento padrão (h1, p, div, etc...) podemos usar outros tipos de seletor para pegarmos elementos específicos no documento HTML.
- Seletor de classe
  - Símbolo do seletor: ponto (.)
  - Seleciona os elementos com o atributo class informado.
- Seletor de id
  - Símbolo do seletor: cerquilha (#)
  - Seleciona o elemento com o atributo id informado.

# Seletores



- Seletor aninhado
  - Seleciona um elemento dentro de outro elemento.
- Seletor universal
  - Símbolo do seletor: asterisco (\*)
  - Seleciona todos os elementos no contexto especificado.



# Exemplos

html\_form\_styled - styles.css

```
1  h1 {
2    color: rgb(1, 53, 105);
3  }
4
5  form label {
6    color: #3a3939;
7  }
8
9  #titulo-home {
10   font-weight: bold;
11 }
12
13 .titulo {
14   font-size: 24px;
15 }
```

html\_form\_styled - styles.css

```
1  * {
2    font-family: 'Courier New';
3  }
4
5  form * {
6    color: #70b193;
7  }
```



# RESET CSS

## Reset CSS

A thick, blue, wavy line that starts from the top right corner and curves downwards and to the left, ending near the top right of the slide.

- Elementos do HTML e navegadores podem ter propriedades CSS por padrão.
- Para que isso não gere inconsistências na visualização da página é importante que seja utilizado essa técnica de resetar o CSS padrão.
- Dessa forma temos total controle sobre o CSS da aplicação.
- As propriedades definidas no reset vão mudar de acordo com a aplicação e a necessidade.

# Exemplo



```
• • •  
  
* {  
  border: 0;  
  box-sizing: border-box;  
}  
  
html, body {  
  margin: 0;  
  padding: 0;  
}  
  
a, button {  
  cursor: pointer;  
}
```



# EXERCÍCIO

## Exercício



Com base na página construída no último exercício, vamos começar a dar vida a ela.

1. Crie um arquivo chamado **styles.css** e incorpore as suas páginas html;
2. Mude a cor do fundo das páginas para a cor *antiquewhite*.
3. Os links da navegação devem ter a cor do texto branca, deve ser removido o sublinhado, o alinhamento do texto deve ser centralizado, o espaçamento interno em todas as direções de 5px e cor de fundo *darkcyan*.
4. Padronize o tamanho das fontes para 12px (menos os títulos (h's)).



# FLEXBOX

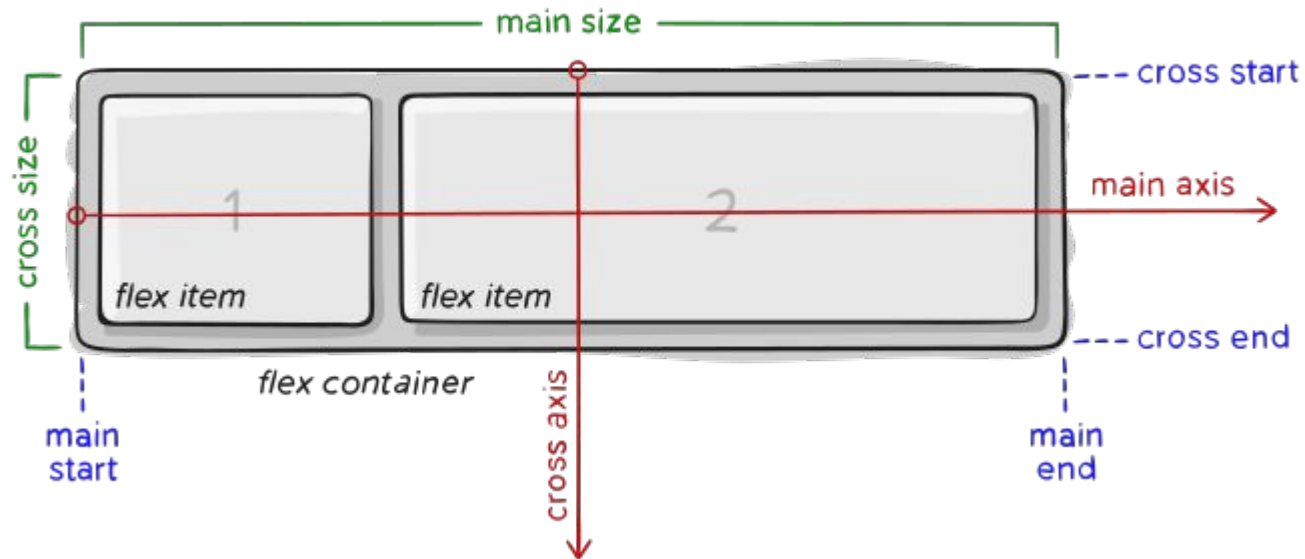
# FlexBox



- Flexible Box Layout.
- Conjunto de propriedades CSS que busca melhorar visualmente a distribuição dos elementos HTML na tela, inclusive para elementos de tamanho dinâmico ou desconhecido.
- Algumas dessas propriedades serão utilizadas no elemento pai, conhecido como **flex container**, e outras nos elementos dentro desse elemento pai, chamados de **flex items**.



# FlexBox



Fonte: <https://css-tricks.com/wp-content/uploads/2018/11/00-basic-terminology.svg>

# Propriedades CSS para Flex Container



- **display: flex**
  - Define que o elemento é um **flex container**, consequentemente, todos os seus elementos-filho são **flex items**.
- **flex-direction: [ row (default) | column | row-reverse | column-reverse]**
  - Define qual eixo será o **eixo principal** do **flex container**.

# Propriedades CSS para Flex Container



- **justify-content:** [ flex-start (default) | flex-end | center | space-between | space-around | space-evenly | ... ]
  - Define a distribuição (alinhamento) dos **flex items** no container no **eixo principal**.
- **align-items:** [ stretch (default) | flex-start | flex-end | center | baseline | ... ]
  - Define a distribuição (alinhamento) dos **flex items** no container no **eixo transversal**.

# Propriedades CSS para Flex Container



- **flex-wrap:** [ nowrap (default) | wrap | wrap-reverse ]
  - Define se o container será multilinhas e em que ordem as linhas devem aparecer.
- **align-content:** [ normal (default) | flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | ... ]
  - Quando os itens no flex container ocuparem mais de uma linha, essa propriedade define o alinhamento entre as linhas.



# Propriedades CSS para Flex Items

- **order: *n*** (default: 0)
  - Define a ordem (peso) que o item deve aparecer no flex container.
- **flex-grow: *n*** (default: 0)
  - Caso o flex container permita que os flex items cresçam, define a proporção que o item crescerá em relação aos outros.
- **align-self: [ auto | flex-start | flex-end | center | baseline | stretch ]**
  - Sobrescreve o comportamento de alinhamento do item definido pelo container na propriedade **align-items**.



# EXERCÍCIO

## Exercício



1. Crie dois quadrados de 200px x 200px, um com o fundo verde e outro com fundo vermelho;
  - a. Alinhe os dois ao centro da tela, tanto verticalmente quanto horizontalmente.
2. Crie uma div com altura de 300px, com largura de 100% da tela e fundo com cor amarela:
  - a. Dentro desta div devem ser adicionados 3 quadrados de 200x200. Dois quadrados devem estar alinhados a esquerda da tela, enquanto o ultimo quadrado deve estar alinhado a direita. Utilize quantas divs forem necessárias para alcançar o comportamento

## Exercício



Para treinar flexbox finalize todos os 24 níveis do Flexbox Froggy (<https://flexboxfroggy.com/#pt-br>).