<LAB365>

HTML

<LAB365> SENAI

AGENDA | M1S02 - A1

- O que é?
- Principais Tags
- Construção página web

O que é HTML

HTML, que significa **HyperText Markup Language (Linguagem de Marcação de Hipertexto)**, é a linguagem padrão usada para **criar e estruturar páginas web**.

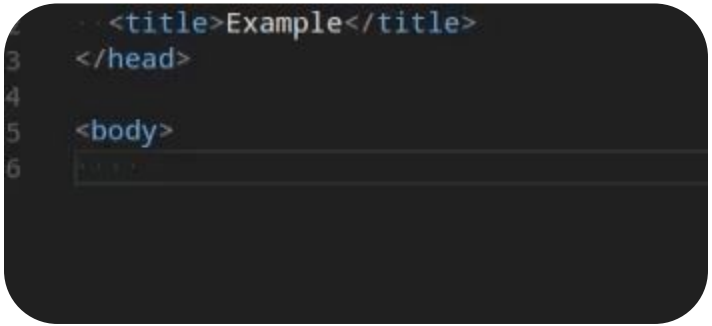
Ele fornece a base para o conteúdo que você vê em navegadores, como texto, imagens, links, formulários, vídeos e outros elementos.

HTML

Acho que uma coisa bem importante quando começamos a programar é entender que toda estrutura de programação normalmente trabalha um conceito de **delimitação** de código e muitas vezes usamos a **indentação** para nos ajudar a ter isso melhor identificável visualmente.

Agora que falamos de Tags HTML é bom lembrar que essas tags podem conter uma tags dentro de outra, se formos trazer isso para o figurativo seria como guardar uma caixa dentro de outra caixa.

Sabendo isso as tags são representadas pelo **<**nomeDaTag**>** e o fechamento da tag adicionamos um **/**, **</**nomeDaTag**>**.



HTML

Vamos ver algumas das principais tags que usamos na estruturação de uma página HTML:

- <!DOCTYPE html>: Define o tipo de documento como HTML5.
- <html>: Envolve todo o conteúdo da página.
- <head>: Contém metadados, como título, links para CSS e scripts.
- <title>: Define o título da página, exibido na aba do navegador.
- <body>: Contém o conteúdo visível da página.
- <h1> a <h6>: Define títulos e subtítulos, onde <h1> é o mais importante e <h6> o menos importante.
- <p>: Define um parágrafo.
-
: Insere uma quebra de linha.
- <hr>: Cria uma linha horizontal para separar conteúdo.
- : Texto em negrito (semântico para importância).
- : Texto em itálico (semântico para ênfase).

HTML

- <a>: Cria um link para outra página ou recurso.
- <img>: Insere uma imagem.
- : Lista não ordenada (com marcadores).
- : Lista ordenada (com números).
- : Item de uma lista.
- <table>: Define a tabela em si. Todos os outros elementos da tabela devem estar dentro desta tag.
- <tr>: Define uma linha da tabela. Cada <tr> contém uma ou mais células (<td> ou <th>).
- <td>: Define uma célula de dados na tabela. Esta tag é usada para conter o conteúdo real da tabela.
- <th>: Define uma célula de cabeçalho. Geralmente é usada na primeira linha ou coluna para descrever o conteúdo das células correspondentes. O texto dentro de <th> é geralmente exibido em negrito e centralizado.

HTML

Nos últimos slides vimos algumas das tags HTML, mais utilizadas. Mas temos várias dela para utilização em diversas necessidades diferentes.

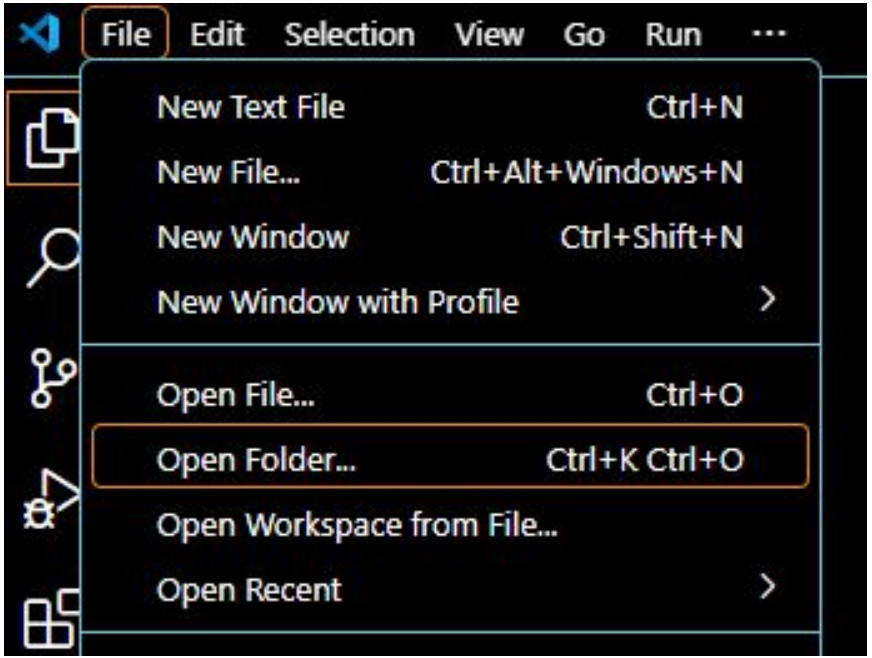
Ao decorrer das aulas nos depararemos com mais tags diferentes, mas podemos ver a lista completa no link a seguir:

https://developer.mozilla.org/pt-BR/docs/Web/HTML/Element

VAMOS CODAR!

O melhor jeito de vermos como essa estrutura funciona é na prática, então vamos criar nossa primeira página web. Para isso vamos abrir o VSCode.

É importante que a primeira coisa a se fazer é criar umas pasta para salvar nosso arquivos, pode criar no local que preferir e abrir a pasta na nossa IDE.



EXEMPLO DE CÓDIGO

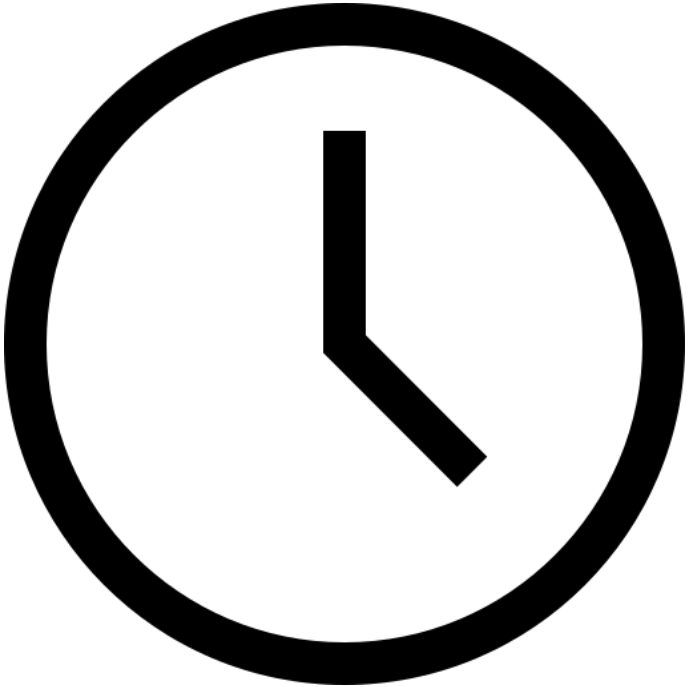
```html
<!DOCTYPE html>
<html lang="pt-BR"></html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Meu Site</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <h1>Bem-vindo ao Meu Site</h1>
        <img src="linkDaimagem.jpg" alt="Imagem de exemplo">
        <ul>
            <li>Item 1</li>
            <li>Item 2</li>
            <li>Item 3</li>
        </ul>
        <form action="/submit" method="post">
            <label for="nome">Nome:</label>
            <input type="text" id="nome" name="nome">
            <label for="email">Email:</label>
            <input type="email" id="email" name="email">
            <button type="submit">Enviar</button>
        </form>
    </header>
    <footer>
        <p>&copy; 2023 Meu Site. Todos os direitos reservados.</p>
    </footer>
</body>
</html>
```

INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20
Retorno: 20:40



HTML

Seguimos agora para construção de uma página com **tabelas**, para isso usaremos as tags que vimos anteriormente e também as tags abaixo:

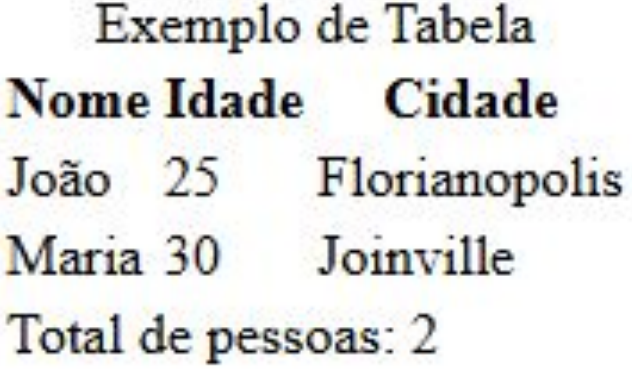
- <thead>: Define o cabeçalho da tabela. Contém um ou mais <tr> com <th> para descrever as colunas ou linhas.
- <tbody>: Define o corpo da tabela. Contém as linhas e células de dados principais.
- <tfoot>: Define o rodapé da tabela. Geralmente usado para resumos ou totais.
- <caption>: Adiciona uma legenda ou título à tabela. Deve ser colocada logo após a tag <table>.

VAMOS CODAR!

Para iniciar nosso código agora vamos utilizar um novo arquivo html, podemos nomear o mesmo como tabela.

Agora aqui dentro vamos ver como podemos utilizar tabelas no HTML.

Uma coisa que gosto de dar ênfase é que estamos lidando **somente com HTML nesse momento**, então nossa tabelas terão linhas invisíveis, na próxima aula começaremos a adicionar detalhes estéticos a nossa formatação (onde poderemos colocar uma cor a linha invisível.



EXEMPLO DE CÓDIGO

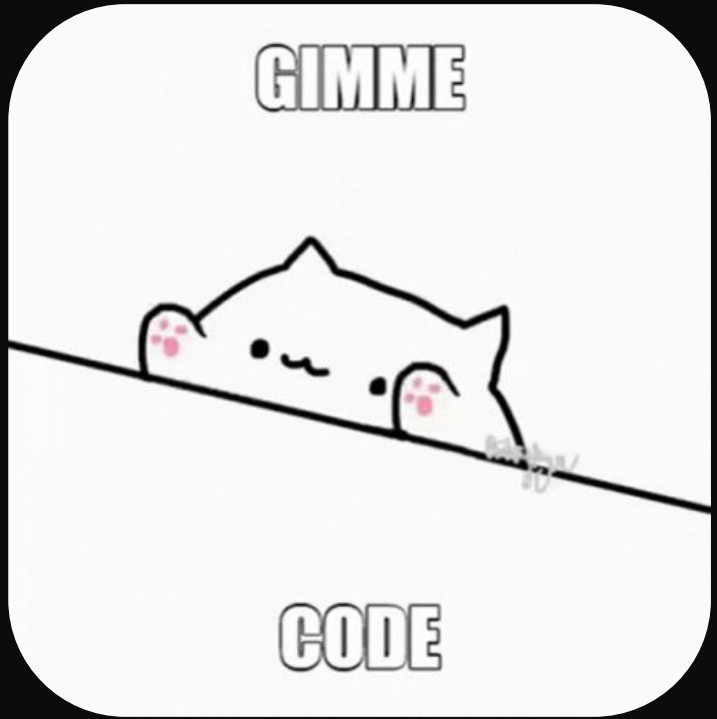
```html
<table>
    <caption>Exemplo de Tabela</caption>
    <thead>
      <tr>
        <th>Nome</th>
        <th>Idade</th>
        <th>Cidade</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>João</td>
        <td>25</td>
        <td>Florianopolis</td>
      </tr>
      <tr>
        <td>Maria</td>
        <td>30</td>
        <td>Joinville</td>
      </tr>
    </tbody>
    <tfoot>
      <tr>
        <td colspan="3">Total de pessoas: 2</td>
      </tr>
    </tfoot>
</table>
```

TREINANDO NOSSAS HABILIDADES!

Como forma de treinar nosso conhecimento adquirido, vamos desenvolver uma página HTML!

Teremos as seguintes necessidades:

- Devemos ter um title;
- Utilizar pelo menos um H1 para título;
- Utilizar uma imagem;
- Possuir algum texto em <p>;
- Possuir uma lista ordenada ou não;
- Possuir uma tabela.



<LAB365>

<LAB365>

CSS

AGENDA | M1S02 - A2

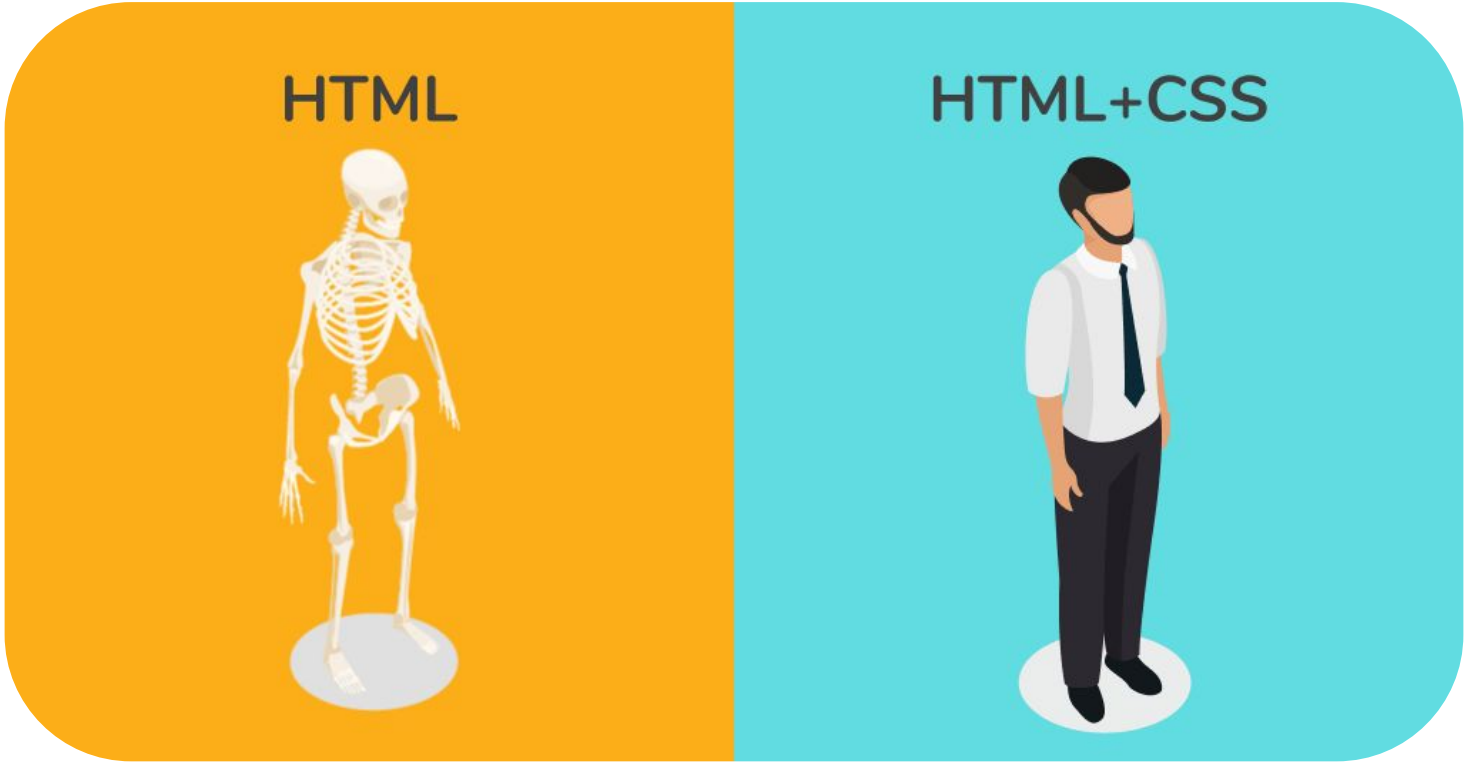
- O que é?
- Seletores
- Responsividade
- Flex Box
- CSS Grid

O que é CSS

CSS, ou **Cascading Style Sheets (Folhas de Estilo em Cascata)**, é uma linguagem usada para **controlar a apresentação e o estilo** de documentos HTML ou XML. Enquanto o **HTML é responsável pela estrutura e conteúdo** de uma página web, o **CSS cuida da aparência, definindo cores, fontes, layouts, espaçamentos e outros aspectos visuais**.

Dentro do CSS temos inúmeras referências, para estar manipulando nossos elementos. Assim mudando cor, tamanho dentre outras propriedades.

O que é CSS



SELETORES

Os seletores do CSS são **padrões** usados para **selecionar e aplicar estilos** a elementos específicos em um documento HTML. Eles são a **base** para definir quais elementos serão afetados pelas regras de estilo.

Juntando nosso conhecimento sobre **seletores e referência**, podemos estar utilizando nossa página web.

https://developer.mozilla.org/pt-BR/docs/Web/CSS/Reference

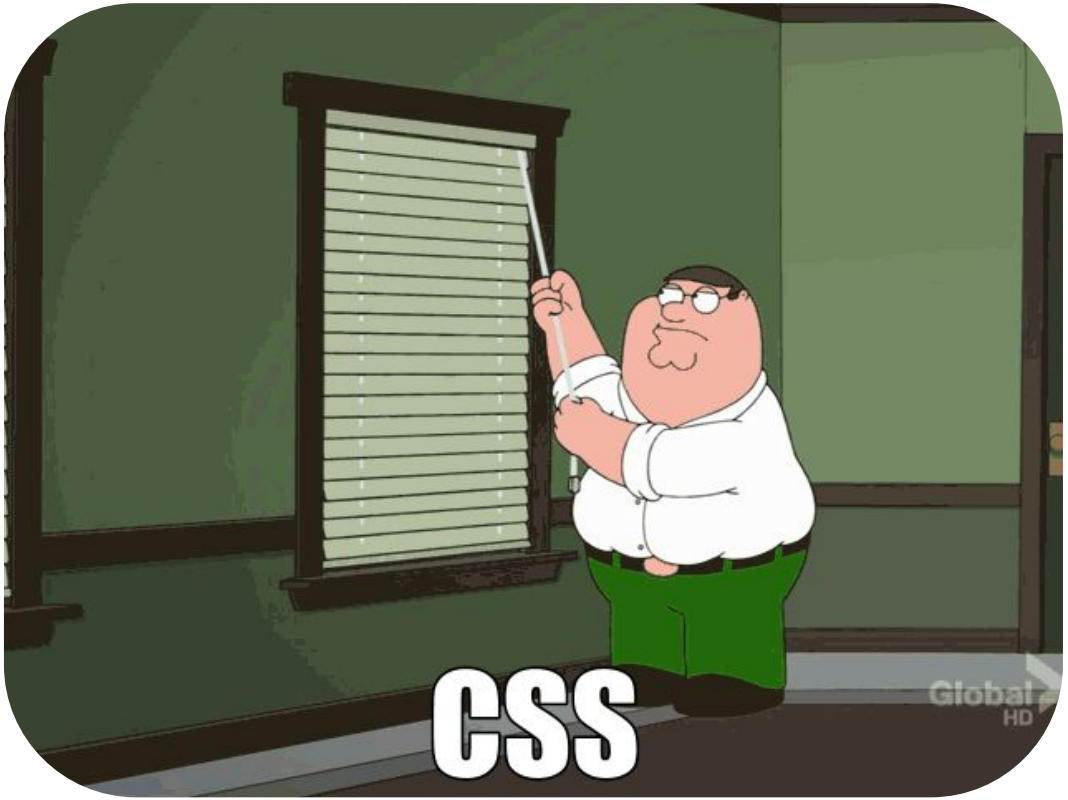
SELETORES

Principais seletores:

- tag: usamos a própria **tag** html (p);
- classe: é usando o . e o nome da classe (.destaque);
- ID: usado o **#** e o nome do id (#titulo);
- universal: usado para selecionar tudo (*);
- Seletor de Descendência: usando para indicar descendência de itens, nesse caso usamos os seletores com o **espaço** ( ) para indicar descendência;

```
p {
    color: blue;
}
.destaque {
    background-color: yellow;
}
#titulo {
    font-size: 24px;
}
* {
    margin: 0;
    padding: 0;
}
div p {
    font-style: italic;
}
```

SELETORES & CSS

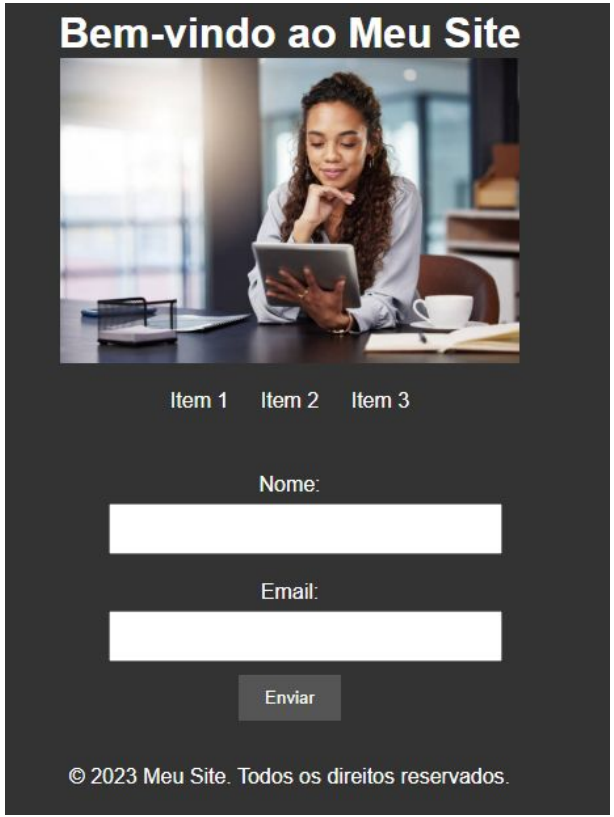


VAMOS CODAR!

A melhor forma de vermos esses conceitos é na aplicação prática e para isso vamos utilizar o código html que já criamos.

Acho que é interessante também sabermos que podemos desenvolver nosso CSS na página HTML dentro da tag <style>, mas a melhor forma de trabalharmos é criando um arquivo CSS e vincular o mesmo com a página HTML.

Assim temos um código mais organizado e de manutenção mais fácil.



EXEMPLO DE CÓDIGO

```css
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
}

header {
    background-color: #333;
    color: white;
    padding: 20px 0;
    text-align: center;
}

header h1 {
    margin: 0;
    display: inline-block;
    vertical-align: middle;
}

header form {
    display: inline-block;
    vertical-align: middle;
    margin-top: 0;
}
```

```css
header img {
    max-width: 100%;
    height: auto;
}

header ul {
    list-style-type: none;
    padding: 0;
}

header ul li {
    display: inline;
    margin: 0 10px;
}

header form {
    margin-top: 20px;
}
```

```css
header form label {
    display: block;
    margin: 10px 0 5px;
}

header form input {
    padding: 10px;
    width: 100%;
    max-width: 300px;
    margin-bottom: 10px;
}

header form button {
    padding: 10px 20px;
    background-color: #555;
    color: white;
    border: none;
    cursor: pointer;
}

header form button:hover {
    background-color: #444;
}
```

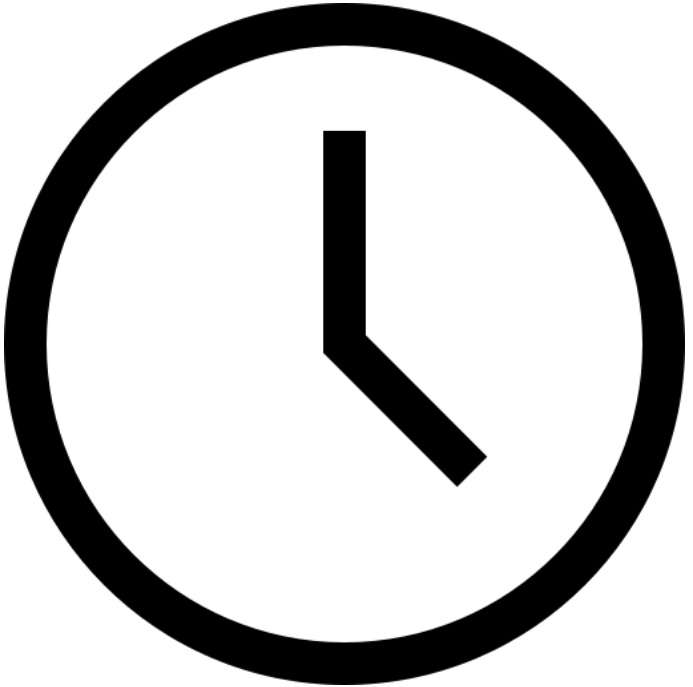
```css
footer {
    background-color: #333;
    color: white;
    text-align: center;
    padding: 10px 0;
    position: fixed;
    width: 100%;
    bottom: 0;
}
```

INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

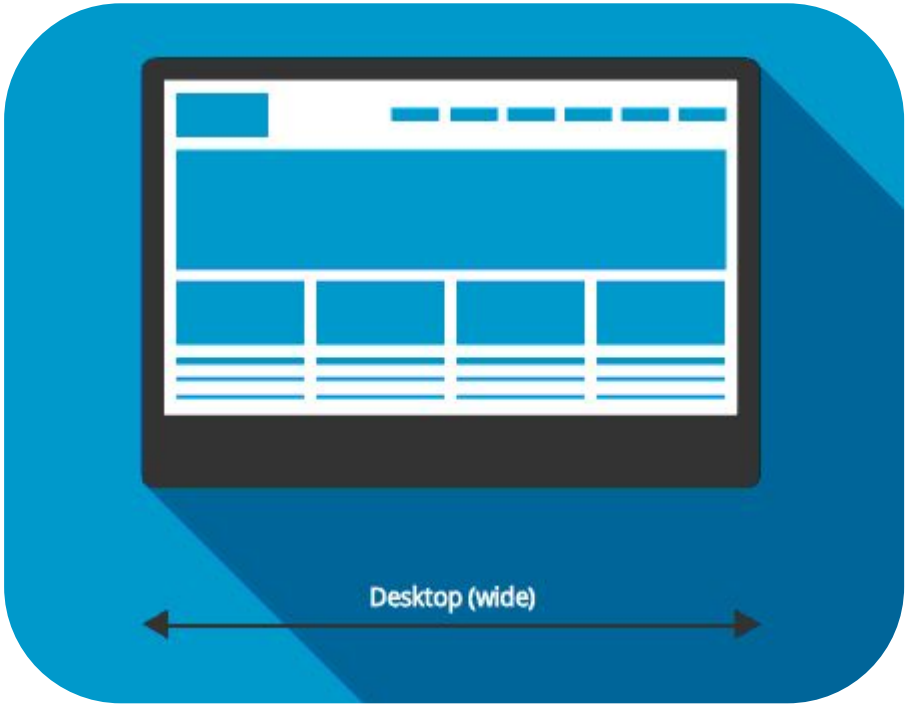
Início: 20:25
Retorno: 20:45



RESPONSIVIDADE

Se trata da capacidade de um site ou aplicação web se **adaptar** a **diferentes tamanhos** de tela e dispositivos, como desktops, tablets e smartphones.

Com o aumento do uso de dispositivos móveis, a responsividade tornou-se um aspecto **essencial** do design e desenvolvimento web.

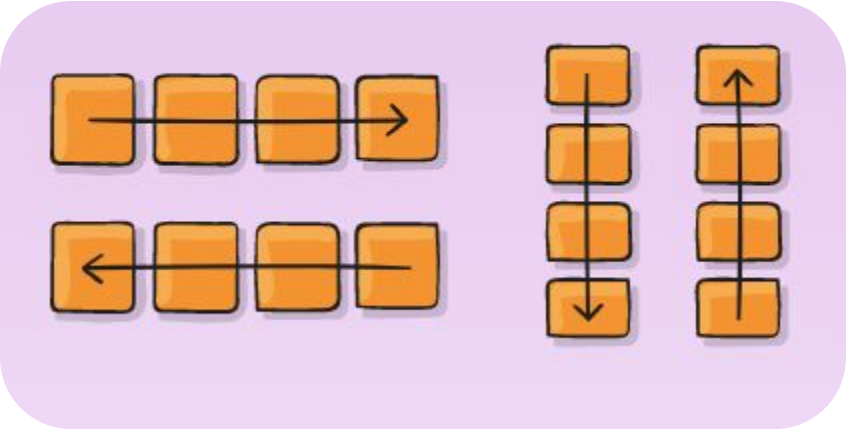


FLEXBOX

O Flexbox é um **modelo de layout unidimensional**, ou seja, ele lida com layouts em uma **única direção** (linha ou coluna) por vez. É ideal para distribuir espaço e alinhar itens dentro de um contêiner.

Quando usar Flexbox?

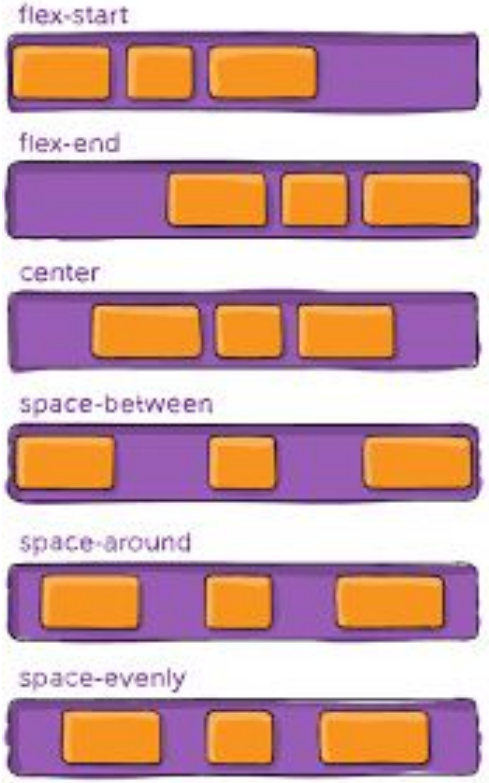
- Para alinhar itens em uma única linha ou coluna;
- Para criar layouts simples, como menus, barras de navegação ou galerias de imagens;
- Para distribuir espaço entre itens de forma dinâmica.



FLEXBOX

Propriedades Principais do Flexbox

- display: flex: Define um contêiner como flexível.
- flex-direction: Define a direção dos itens (row, row-reverse, column, column-reverse).
- justify-content: Alinha os itens ao longo do eixo principal (flex-start, flex-end, center, space-between, space-around, space-evenly).
- align-items: Alinha os itens ao longo do eixo transversal (flex-start, flex-end, center, stretch, baseline).
- flex-wrap: Controla se os itens devem quebrar para uma nova linha (nowrap, wrap, wrap-reverse).
- align-content: Alinha as linhas de itens ao longo do eixo transversal (quando há múltiplas linhas).



EXEMPLO DE CÓDIGO

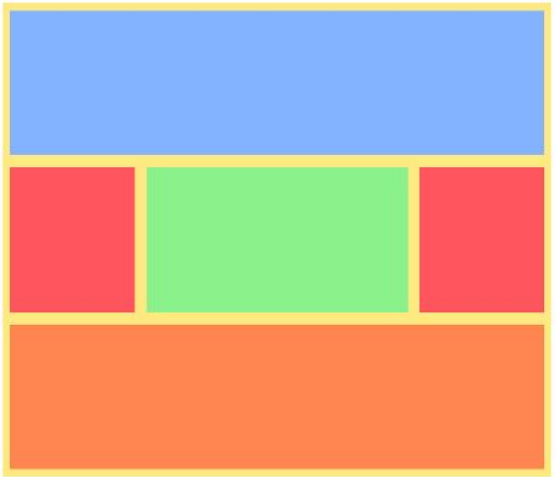
```html
<!DOCTYPE html>
<html lang="pt-BR"></html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ex</title>
    <link rel="stylesheet" href="stylesflexEx.css">
</head>
<body>
    <div class="container">
        <div class="item">1</div>
        <div class="item">2</div>
        <div class="item">3</div>
    </div>
</body>
</html>
```

EXEMPLO DE CÓDIGO

```css
.container {
    display: flex;
    justify-content: space-between; /* Distribui o espaço entre os itens */
    align-items: center; /* Centraliza verticalmente */
    background-color: #f4f4f4;
    padding: 10px;
}
.item {
    background-color: #333;
    color: white;
    padding: 20px;
    margin: 5px;
}
```

CSS GRID

O CSS Grid é um **modelo de layout bidimensional**, ou seja, ele permite criar layouts complexos com **linhas e colunas**. É ideal para designs que precisam de controle preciso sobre a posição dos elementos.



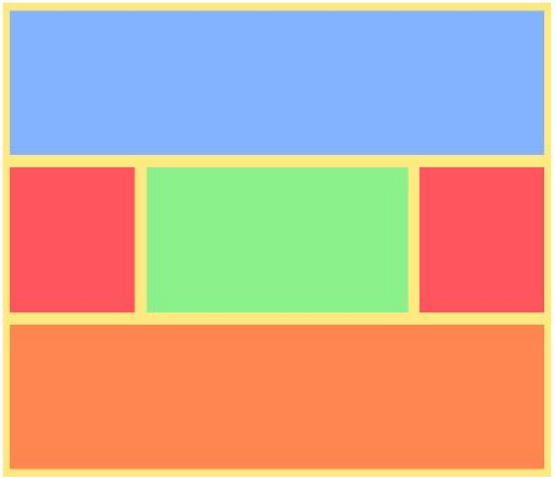
Quando usar CSS Grid?

- Para criar layouts complexos com múltiplas linhas e colunas;
- Para designs que precisam de alinhamento preciso em ambos os eixos (horizontal e vertical);
- Para layouts de grade, como galerias de imagens, painéis de administração ou designs de revista.

CSS GRID

Propriedades Principais do CSS Grid

- display: grid: Define um contêiner como grid.
- grid-template-columns: Define o número e o tamanho das colunas.
- grid-template-rows: Define o número e o tamanho das linhas.
- gap: Define o espaçamento entre as células da grade (row-gap e column-gap).
- justify-items: Alinha os itens ao longo do eixo horizontal dentro das células.
- align-items: Alinha os itens ao longo do eixo vertical dentro das células.
- grid-column e grid-row: Define a posição de um item na grade.



EXEMPLO DE CÓDIGO

```html
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Layout com CSS Grid</title>
    <link rel="stylesheet" href="stylesCssGridEx.css">
</head>
<body>
    <div class="container">
        <header class="header">Cabeçalho</header>
        <nav class="nav">Barra Lateral</nav>
        <main class="main">Conteúdo Principal</main>
        <aside class="sidebar">Sidebar</aside>
        <footer class="footer">Rodapé</footer>
    </div>
</body>
</html>
```

EXEMPLO DE CÓDIGO

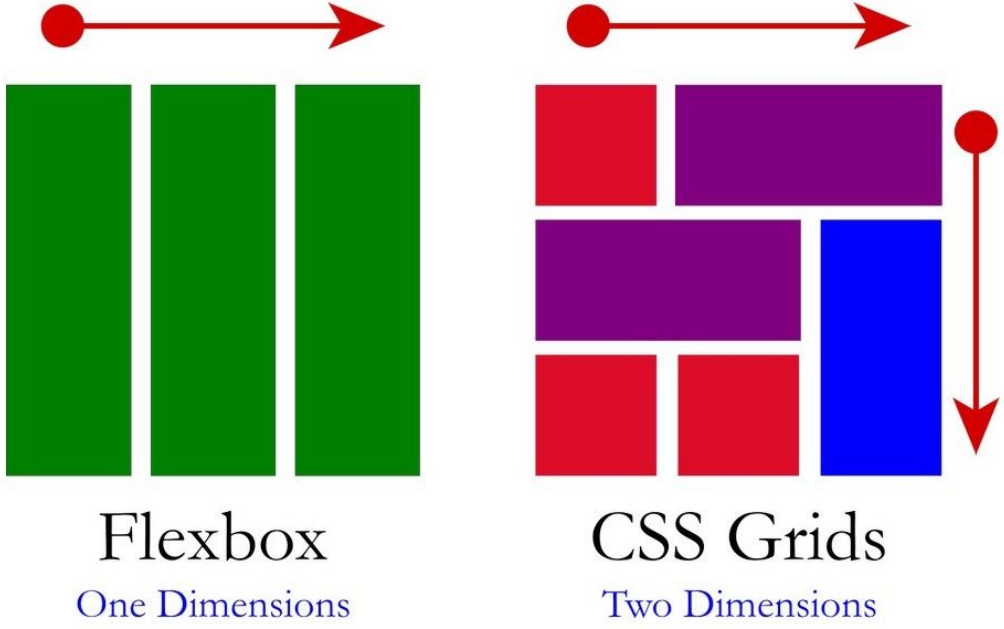
```css
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: Arial, sans-serif;
    line-height: 1.6;
}

.container {
    display: grid;
    grid-template-columns: 1fr 3fr 1fr; /* 3 colunas: sidebar | conteúdo | sidebar */
    grid-template-rows: auto 1fr auto; /* 3 linhas: cabeçalho | conteúdo | rodapé */
    gap: 10px; /* Espaçamento entre as áreas */
    height: 100vh; /* Ocupa toda a altura da tela */
    padding: 10px;
}

.header {
    grid-column: 1 / -1; /* Ocupa todas as colunas */
    background-color: #333;
    color: white;
    padding: 20px;
    text-align: center;
}
```

```css
.nav {
    background-color: #f4f4f4;
    padding: 20px;
}

.main {
    background-color: #e0e0e0;
    padding: 20px;
}

.sidebar {
    background-color: #f4f4f4;
    padding: 20px;
}

.footer {
    grid-column: 1 / -1; /* Ocupa todas as colunas */
    background-color: #333;
    color: white;
    padding: 20px;
    text-align: center;
}
```

CSS GRID

Aqui vimos um exemplo um pouco mais complexo da utilização do grid css para criar uma esqueleto do que poderia ser uma página web.

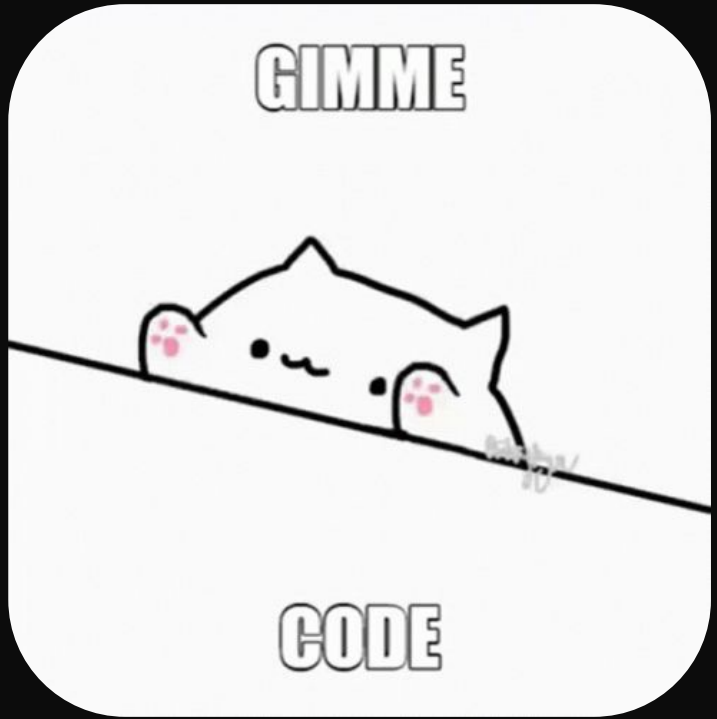
Obviamente podemos trabalhar utilizando somente o CSS Grid ou o Flex box, como também podemos mesclar o uso dos dois... Para assim atender nossas necessidades.



TREINANDO NOSSAS HABILIDADES!

Como forma de treinar nosso conhecimento adquirido, vocês devem criar um layout usando CSS Grid/Flexbox (um ou ambos) para a página HTML que criaram no exercício anterior.

Na próxima aula veremos os layouts dessa página, como cada um personalizou a mesma, então após criar já separe uma imagem de como ficou a visualização de sua página.



<LAB365>

<LAB365>

Revisão

<LAB365> SENAI

AGENDA | M1S02 - A3

- **Um pouco mais de CSS Grid!**
- Verificando Layouts
- Revisão HTML e CSS
- Correção de exercícios
- Avaliação do docente

VAMOS VER OS LAYOUTS!

Vamos usar o Padlet!

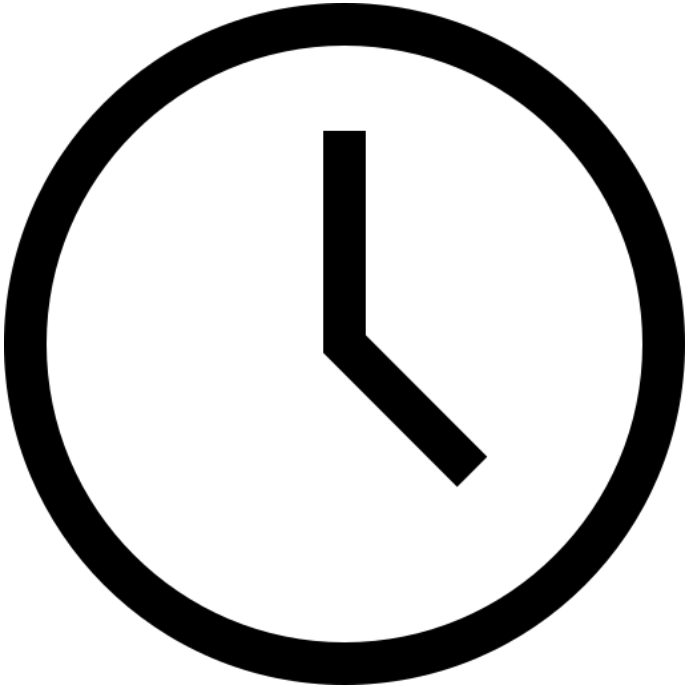
https://padlet.com/ottohannemann/meu-padlet-iluminado-z487sqeukczz17iz

INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

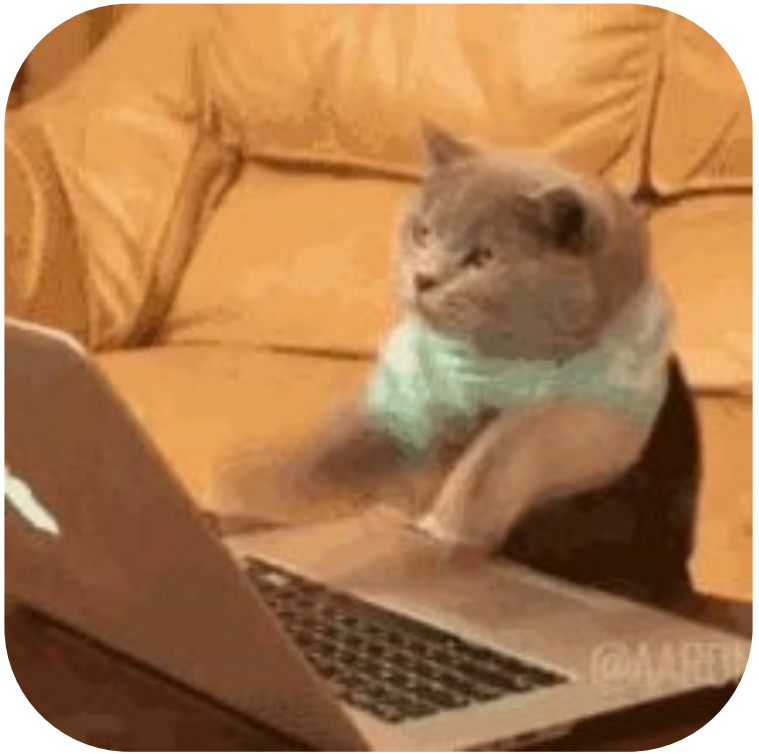
Início: 20:30
Retorno: 20:50



REVISÃO E CORREÇÕES

Nossa revisão será programando!

- HTML
  - Tags
- CSS
  - Seletores e referências
  - FlexBox
  - CSS Grid



AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique aqui ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.

<LAB365>