**HTML**

**<h1> -- abertura de tag**

**</h1> -- fechamento de tag**

**<img src="foto.png" alt="Exemplo de foto">**

**<img... -- abertura de tag**

**"src= e alt=" -- parâmetro**

**Conteúdo entre "" -- valor**

**Estilo em CSS**

**h1 {**

**font-family: Arial;**

**font-size: 20pt;**

**color: blue;**

**}**

**h1 -- Seletor**

**font-size: 20pt; -- declaração**

**declaração = propriedade + valor**

**font size = propriedade e blue = valor**

**h1 = título principal**

**h2 = secundário**

**h3 = ....**

**até h6**

**Não obrigatoriamente h1 é grande e h6 é pequeno**

**Podemos ter mais de um h1 por página**

**FAVICON - <link> - escolher favicon**

**ctrl + shift + P , abb , Wrap with abbreviation = para colocar uma tag no início e final de um trecho selecionado**

**--- Mais rápido: ctrl+shift+P, Enter, <tag desejada>, Enter**

**tag <pre> mantém a formatação**

**tag <code> muda os caracteres para um mono espaçado**

**Recomendado utilizar a tag <q> (quote) para citações, ao invés de apenas digitar ""**

**tag <blockquote> para citações complexas e maiores**

**- <blockquote cite="link"**

**tag <abbr> para inserir o significado de siglas. Ex: HTML**

**<bdo> para textos invertidos**

**- <bdo dir="rtl"> Texto </bdo>**

**INÚTIL AHAHAHA**

**<ol> ordered list**

**<li> itens da lista**

**</li> opcional a partir do HTML5**

**<ul> unordered list**

**<a> âncora - links externos**

**target="\_blank" - para abrir numa nova aba**

**target="\_self" - para abrir na mesma aba**

**rel="external" - para melhor indexação**

**rel="nofollow" - para links externos onde você não dá um "aval" (propagandas, etc)**

**rel="next" - para página secundária dentro do seu site**

**rel="prev" - para retornar à home**

**Quando for criar vários links ou uma lista, ao invés de colocar a tag várias vezes, é só escrever (tag\*número de vezes) e dar enter.**

**Por exemplo:**

**a\*5 = 5 links**

**li\*10 = 10 elementos da lista.**

**DOWNLOADS**

**<a href="endereço\_do\_arquivo" download="nome\_do\_arquivo"> texto a ser clicado aqui </a>**

**Para baixar direto sem antes abrir o arquivo, incluir também o parâmetro type="ler abaixo"**

**iana.org/assignments/media-types**

**IMAGEM DINÂNICA**

**tag <picture> - possibilia vários src (fontes)**

**Dentro dela usar a <source> (media type) pra cada imagem, da menor pra maior (lembrar de preencher com a extensão em type="image/..."**

**por ultimo usar a tag <img> para a maior imagem**

**Ex:**

**<picture>**

**<source media="(max-width: 750px)" srcset="imagens/foto-p.png" type="image/png">**

**<source media="(max-width: 1050px)" srcset="imagens/foto-m.png" type="image/png">**

**<img src="imagens/foto-g.png" alt="imagem flexivel">**

**</picture>**

**Mudar a ordem pode quebrar a função de imagem dinâmica**

**ÁUDIOS**

**Forma 1:**

**Tag <audio>, "controls" dentro da tag audio**

**- Se quiser que toque automaticamente a iniciar, utilizar "autoplay" dentro da tag <audio>**

**- Se quiser que o audio toque em looping, utilizar "loop" dentro da tag <audio>**

**- Compatível com .mp3 / .wav / .ogg (evitar o wav pois são mais pesados)**

**Forma 2:**

**Entre <audio> e </audio> utilizar a <source src="" type="audio/">**

**--- no caso de mp3 usar o type="audio/mpeg". wav e ogg não mudam**

**Dentro da <audio> usar preload="metadata" controls**

**VÍDEOS**

**Ex1:**

**<video src="video.mp4" width="800" poster="" controls autoplay loop></video>**

**- poster="" para utilizar capa no primeiro frame**

**Ex2:**

**<video width="320" height="240" controls>**

**<source src="movie.mp4" type="video/mp4">**

**<source src="movie.ogg" type="video/ogg">**

**Your browser does not support the video tag.**

**</video>**

**- Compatível com .mp4 / .webm (qualidade piorada) / .ogv**

**Programa HandBrake para conversão**

**CSS**

**INLINE - é feito dentro de cada tag, o que é trabalhoso e não recomendado. Deve ser utilizado em casos e ajustes bem pontuais.**

**Ex:**

**<body style="backgroud-color: black; font-family: arial"**

**<h1 style="color: white;">**

**INTERNOS - é utilizado com a tag <style> dentro da tag <head> e pode ser útil para sites mais simples, com uma ou poucas páginas.**

**Ex:**

**<style>**

**h1 {**

**background-color: navy;**

**}**

**</style>**

**EXTERNOS - o mais prático, utilizado para mudar os estilos de todas as páginas de uma vez. Utilizar a tag <link> (link:css) dentro da tag <head> para criar a página "nomedapagina.css"**

**--- Criar a @charset "UTF-8"; dentro da página de CSS**

**--- Escrever dentro da página de CSS da mesma forma que escreve dentro da tag <style> </style>**

**Colorzilla** para selecionares cores de sites

**@import** deve sempre ficar no início da tag <style> ou no início do documento CSS externo e FORA de um seletor {}

**FONT-FACE**

@font-face {

font-family: "nome que você quiser";

src: url(selecionar a pasta e arquivo);

}

Chrome Web Store

- **Fonts Ninja** - extensão para reconhecer fontes nos sites, tamanho, espaçamento, etc

**FONTES EM IMAGENS**

- WhatFontIs.com

- Fontsquirrel.com

- MyFonts.com

**ALINHAMENTO**

- Será considerado o alinhamento que mais específico (body, h1, etc)

text-indent: 30px; - para início de parágrafos (espaçamento)

**SELETORES**

**# = id**

**. = class**

**: = pseudo-class**

**:: = pseudo-element**

**> = children**

Em HTML é id – em CSS é # (utilizar em apenas um elemento)

Ex: <h1 id=”destaque”> / #destaque { } ou h1#destaque { }

Em HTML é class – em CSS é . (pode utilizar em vários elementos)

Ex: <h1 class=”destaque”> / .destaque { }

Obs: um elemento pode conter configurações de id e de class ao mesmo tempo, mas as do id sobrepõem-se às de class. Ex: <h1 id=”titulo” class=”destaque”>

Também é possível colocar mais de uma class em um mesmo elemento. Ex: <h1 class=”titulo destaque”>

**PSEUDO-CLASSES = :**

Uma ***pseudo-classe*** CSS é uma palavra-chave adicionada a seletores que especifica um estado especial do elemento selecionado. Por exemplo, [:hover](https://developer.mozilla.org/pt-BR/docs/Web/CSS/:hover) pode ser usado para alterar a cor de um botão quando o usuário passar o cursor sobre ele.

Ex:

HTML -- <h1 class=”titulo”> TESTE </h1>

CSS -- .titulo {

background-color: black;

}

.titulo:hover {

Background-color: yellor;

}

:visited – muda aparência de links já visitados/clicados

:active – muda a aparência com a ação do clique do mouse

**PSEUDO-ELEMENTOS em CSS = ::**

Um **pseudo-elemento** CSS é uma palavra-chave adicionada a um seletor que permite que você estilize uma parte específica do elemento selecionado. Por exemplo, o pseudo-elemento [::first-line](https://developer.mozilla.org/pt-BR/docs/Web/CSS/::first-line) aplica o estilo apenas na primeira linha de um parágrafo.

::before – adiciona algo antes do elemento selecionado

::after - adiciona algo após o elemento selecionado

Ex:

h1::before {

content: “»”;

}

h1::after {

content: “«”;

}

**MODELO DE CAIXAS**

Aninhamento – uma caixa dentro de outra

**Height** – altura

**Width** – largura

**Border** – colado no conteúdo

**Padding** – espaço entre border e o conteúdo

**Margin** – espaço para fora do conteúdo, após a borda

**Outline** – uma nova borda, fora do elemento e colado à borda

**Box-level** – quebra o conteúdo, preenche a largura inteira na próxima linha, quebra de novo e retoma o conteúdo. Exemplos: <div>, <h1> ~<h6>, <p>, <main>, <header>...

**Inline-level** – continua na mesma linha onde estava o conteúdo e também retoma o conteúdo na mesma linha. Exemplo: <span>, <a>, <code>, <small>, <strong>, <sup/ sub>...

**FAZER NO INÍCIO DO CSS**

Pra quem não quiser sofrer com erro de margins e paddings excessivos na página, só criar em todo projeto um: \* { margin: 0; padding: 0; border: 0; }

**SHORTHAND de estilização**: largura/espessura estilo cor

Ex: border: 5px solid red

**GROUPING TAGS** (tags de agrupamento)

**Header**

**Main** Todos são vistos como <**div**> mas os navegadores conseguem indexar melhor e dar mais relevância dessa forma

**Footer**

**Aside** - deve ser usada para criar uma seção em uma página web com conteúdo não necessariamente relacionado com o principal, algo totalmente separado do contexto. Um ótimo exemplo da utilização de <aside> é para criar uma barra lateral com propagandas, conhecida como sidebar. Além disso, pode ser usada para biografia do autor, grupos de elementos de navegação externa, pull quotes e outros conteúdos considerados separados do principal da página.

**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. O principal objetivo é retirar essa responsabilidade das divs. Tendo como principal diferencial a navegação semântica com HTML 5.

Digamos que na sua página inicial você tenha três chamadas principais: Clientes, Serviços e Notícias. Aí um ótimo lugar para você usar a tag section. Cada uma dessas chamadas podem ser um elemento de seção. A tag pode ser usada em conjunto com o H1, H2, H3, H4, H5 e H6, elementos para indicar a estrutura do documento. Ele pode conter um header e um footer se necessário.

**BORDAS PERSONALIZADAS**

Utilizar normalmente o **border (solid ou transparent)** e em seguida:

Utilizar border-image-source: url(‘nomedoarquivo’);

border-image-slice: “número”;

border-image-repeat: repeat;

**SHORT HAND** – border-image: url(“nomedoarquivo”) 27 repeat;

**PLANEJAMENTO DE SITE**

**Mockflow.com** – criar conta gratuita

Fazer um rascunho, imaginar a diagramação do site que irá criar

Além do main e footer, lembrar de utilizar:

Nav (menu) article, aside,

**Variáveis em CSS para facilitar a criação do projeto:**

:root {

(sempre começar com –)

--cor0: “cor”;

--cor1: “cor”;

--cor3: “cor”;

--fonte-padrao: Arial, Verdana, ...;

--fonte-destaque: “Bebas Neue”, cursive;

}

Ex:

h1 {

font-family: var(--fonte-titulo);

color: var(--cor1);

**Git** – repositório local, versionamento de software

**Github** – repositório remoto e também hospedagem

**Push** – enviar atualizações do local para o remoto

**GitHub em vários pcs**

- Fazer login no github desktop

- Depois vai em File > Options > Git- clica nos dois campos e depois em save

- Escolhe qual repositório quer clonar e depois clica em clonar

-

Criei o exercício 022

- DISPLAY: INLINE-BLOCK - Deixa tudo na mesma linha

- background-size: (px, %, etc)

- **contain**: exibirá a imagem toda, mas criando espaços dos lados ou acima/abaixo da tela, como nos filmes, dependendo das dimensões da tela;

- **cover**: exibirá a imagem toda, mas cortando partes da imagem de acordo com o tamanho da viewport;

- 100% (base/largura) 100% (altura): exibirá em todo o espaço selecionado, desconsiderando a proporcionalidade;

- background-repeat: no-repeat, repeat-x (linha), repeat-y (coluna)

- background-position: left top (padrão), left center, right bottom, center center, etc - **PRIMEIRO AS COLUNAS E DEPOIS AS LINHAS**

**ATENÇÃO**: no height utilizar 97vh (98, 99, etc), onde **vh** = a 1/100 da altura da viewport. Então, por exemplo, se a altura do navegador é 900px, 1vh equivale a 9px

**DICA:** Configurar cor de fundo do site para que o usuário já possa consumir o conteúdo antes da página carregar por completo, vídeos, imagens grandes, etc.

background-attachment: scroll (padrão, rola junto), fixed (fixo),

**SHORTHAND - BACKGROUND**: color > image > position >/> size > repeat > attachment

**ATALHO PARA COMENTÁRIOS: Selecione a palavra e depois pressione Shift + Alt + A**

**ALINHAMENTO VERTICAL**

Display: block

Margin: auto

**ALINHAMENTO VERTICAL**

position:

Quando A é pai de B, então colocar relative para o A e absolute para o B

**A = relative**

**B = absolute**

Ao utilizar o absolute num elemento, você libera a utilização dos elementos left e top. Ex:

Left: 50%; (px, etc)

Top: 50%;

**PARA CENTRALIZAR, ADICIONAR:**

Transform: translate(-50%, -50%)

display: flex; - Possui diversas funções (procurem pela documentação), uma delas é habilitar o manejamento do conteúdo

justify-content: center; - Alinhamento HORIZONTAL, ou seja, dispensa o "margin: auto"

align-items: center; - Alinhamento vertical = text-align: center

**font-variant: small-caps** – todas ficam em forma de caixa alta, mas as iniciais permanecem maiores.

**UTILIZAR MAIS ID JUNTO COM CLASS**

Ex:

<section class=”boximagem” id=”img01”>

<section class=”boximagem” id=”img02”>

.boximagem {

color: #FFF;

}

.boximagem#img01 {

Background-image: url (../imagens/img01.jpg)

}

.boximagem#img02 {

Background-image: url (../imagens/img02.jpg)

}

**TABELAS**

Tabelas ainda são úteis, dependendo da sua utilização. Utilizar tabelas para a criação da estrutura de sites, por exemplo, já caiu em desuso.

Hierarquia de tabelas (simples)

TABLE = tabela = <table>

TABLE ROW = linha de tabela = <tr>

TABLE HEADER = cabeçalho de tabela = <th>

<th **scope**=”row” ou “col”> para definir do que ele é título, da linha ou coluna. No caso de conjunto de linhas ou colunas, **rowgroup** ou **colgroup**.

TABLE DATA = dado de tabela = <td>

OBS: Não é obrigatório o fechamento dessas tags (</tr>, </td>, etc), porém requer cuidado e atenção. Caso não respeite a hierarquia correta, o conteúdo poderá aparecer antes ou depois da tabela.

**Border-collapse**: separete (padrão, um pouco separadas), colapse (juntas).

**Vertical-align**: alinhamento vertical para células da tabela – top, middle, bottom

**<CAPTION>**:utilizar antes da <thead>. É a legenda/título da tabela e aparece logo acima, mas fora da tabela

**EFEITO ZEBRADO**

tbody > tr:nth-child(even/odd/2n/2n-1) {

            background-color: lightgrey;

        }

**CONGELAMENTO DO TÍTULO/CABEÇALHO**

table {

            position: relative;

        }

thead {

            position: sticky;

            top: -1px;

        }

**MESCLAGEM DE CÉLULAS**

Existem expansões em forma de coluna e em forma de linha. Ex:

**Coluna**

<td colspan="2">B</td>

2 = número de colunas que B ocupará

**Linha**

<td rowspan="3">D</td>

3 = número de linhas que D ocupará

**AGRUPAMENTO DE COLUNAS**

É possível criar um grupo de colunas para facilitar a sua personalização em conjunto com **colgroup**. Ex:

<style>

        .cnome {

            background-color: aquamarine;

        }

        .cgrupo {

            background-color: blueviolet;

        }

    </style>

</head>

<body>

    <table>

        <colgroup> - tag para criar os agrupamentos

            <col class="cnome"> - usar um <col> para cada coluna

            <col class="cgrupo" span="3"> - span seleciona outras colunas

        </colgroup>

**TABELAS RESPONSIVAS**

Para que tabelas não criem o scroll em toda a página, é possível criar o scroll apenas para a própria tabela. Ex:

<style>

        #container {

            overflow-x: auto; - x para eixo horizontal e y para o vertical, ou utilizar apenas “overflow”.

        }

    </style>

</head>

<body>

    <div id="container"> - Envelopar a tabela numa div e classificar a div com uma id ou class.

        <table>

**IFRAMES**

Mostra um quadro com uma prévia do site desejado, sendo possível personalizar o quadro.

Recomenda-se utilizar um link do site “por trás”, dentro do iframe, para caso o navegador não suporte a função. Ex:

    <p>Acesando o site do

        <iframe src="https://www.cursoemvideo.com" frameborder="0">

            <a href="https://www.cursoemvideo.com" target="\_blank" rel="external">Curso em Vídeo</a>

        </iframe>

        para aprender a programar.</p>

O tamanho padrão dos iframes é de **300x150**, sendo possível personaliza-los dentro da própria tag ou via CSS\*.

Ex1:

<p>Acesando o site do

        <iframe src="https://www.cursoemvideo.com" frameborder="0" width="300" height="300">

            <a href="https://www.cursoemvideo.com" target="\_blank" rel="external">Curso em Vídeo</a>

        </iframe>

        para aprender a programar.</p>

    <p>O site do

Ex2:

<style>

        iframe {

            width: 200px;

            height: 200px;

        }

    </style>

\*O CSS é soberano e sempre prevalecerá sobre as demais personalizações.

Parâmetro **scrolling=”no”** faz com que suma a barra de rolagem na prévia do site

Para personalizar a borda do iframe, marcar como 0 dentro da tag no html e personalizar normalmente via CSS.

É possível carregar páginas/documentos locais dentro dos iframes. Ex:

<ul>

        <li><a href="paginas-extras/img001.html" target="frame">Primeira página</a></li>

        <li><a href="paginas-extras/img002.html" target="frame">Segunda página</a></li>

        <li><a href="paginas-extras/img003.html" target="frame">Terceira página</a></li>

    </ul>

<iframe src="" name="frame">

        <p>Perdão, o seu navegador não suporta esta função.</p>

    </iframe>

Para que inicialmente o iframe não fique vazio, você pode já carregar uma página pelo src normalmente.

**Inconvenientes do iframe**

- Para conteúdos muito importantes, evitar o uso de iframes (indexação);

- Acessibilidade e usabilidade: alguns leitores de tela (para pessoas cegas) não integram muito bem com os iframes. Alguns navegadores também têm dificuldade em reconhecer comandos, como o de voltar (site principal ou o de dentro do iframe?), target blank, etc;

**Tornando iframes mais seguros**

- Para não permitir cadastros ou solicitações de informações dos usuários:

**1) sandbox**

    <iframe src="paginas-extras/pag004.html" sandbox="sandbox">

        Infelizmente o seu navegador não suporta essa função.

    </iframe>

É possível conceder algumas permissões a formulários específicos, utilizando, ao invés de “sandbox”, **“allow-same-origin allow-forms allow-scripts”**

**2) referrerpolicy**

    <iframe src="paginas-extras/pag004.html" sandbox="sandbox" referrerpolicy="no-referrer">

        Infelizmente o seu navegador não suporta essa função.

    </iframe>

**EFEITO ÍCONE**

a > img {

    height: 7vw;

    width: 7vw;

    max-height: 50px;

    max-width: 50px;

    margin: 10px;

    border-radius: 50%;

    border: 1px solid transparent; - já usar uma borda transparente para evitar “tremer” OU “box-sizing: border-box;”

    box-shadow: 2px 2px 5px rgb(0, 0, 0);

}

a > img:hover {

    border: 1px solid white; - mudra a cor da sombra

    margin: 5px; - diminuir margin

    margin-bottom: 15px; - somar os dois margins e utilizar o valor para a margin-bottom

    transition: 0.5s;

    box-shadow: 7px 7px 10px rgb(0, 0, 0); - aumentar a sombra

}

OU

section[#redes](https://www.youtube.com/hashtag/redes)-sociais img:active { transform: scale(95%); }

OU

No hover, usar “transform: translate(-3px, -3px)” além de aumentar a sombra

**FORMULÁRIOS**

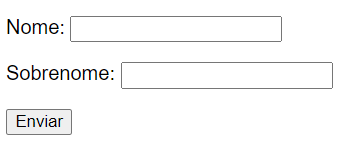
<form>

        <p>Nome: <input type="text" name="Nome" id="Nome"></p>

        <p>Sobrenome: <input type="text" name="Sobrenome" id="Sobrenome"></p>

        <p><input type="submit" value="Enviar"></p>

    </form>



<form autocomplete="on / off"> para ligar ou desligar a sugestão

<form action="cadastro.php"> para informar para onde esses dados serão enviados

**Labels:** usadas para criar o vínculo entre o texto e o campo de preenchimento do formulário (objeto e etiqueta).

<form action="cadastro.php" autocomplete="off">

        <p><label for="inome">Nome:</label> <input type="text" name="nome" id="inome"></p>

        <p><input type="submit" value="Enviar"></p>

    </form>

“**name**” é mais útil para linguagens como **HTML e PHP**

“**id**” é mais útil para linguagens como o **JavaScript**

**Métodos GET e POST**

GET: o dados aparecem na url, pode ser utilizado para dados não sensíveis ou quando você deseja um formulário que gere link compartilhável. Dados com GET devem possuir até 3.000 bytes (aproximadamente 3 mil caracteres).

POST: utilizar para o caso de dados sensíveis (**o ideal é criptografia https**), dados com mais de 3.000 bytes, envio de arquivos (fotos, vídeos, etc).

<form action="cadastro.php" method="get / post">

**Caixas de texto e de senha**

Dentro de **<input>** utilizar **required** para tornar obrigatório o preenchimento do respectivo campo:

<p>

<label for="isenha">Senha:</label> <input type="password" name="senha" id="isenha" required>

</p>

Para especificar o mínimo e/ou o máximo de caracteres utilizar, respectivamente, **minlength** e **maxlength**:

<p>

<label for="isenha">Senha:</label> <input type="password" name="senha" id="isenha" required minlength="8" maxlength="16">

</p>

O **size** ajusta o tamanho da caixa, impactando em quantas letras serão mostradas por vez, mesmo que mais sejam aceitas:

<p>

<label for="inome">Usuário:</label> <input type="text" name="nome" id="inome" required minlength="5" maxlength="16" size="7">

</p>

O **placeholder** ajuda o usuário com uma dica dentro da caixa:

<p>

<label for="inome">Usuário:</label> <input type="text" name="nome" id="inome" required minlength="5" maxlength="16" size="14" placeholder="digite seu usuário">

</p>

O **autocomplete** além de ser utilizado dentro do **<form>** também pode ser utilizado dentro do **<input>** para ajudar os navegadores a saberem quais dados devem completar:

<p>

<label for="inome">Usuário:</label> <input type="text" name="nome" id="inome" required minlength="5" maxlength="16" size="14" placeholder="digite seu usuário" autocomplete="username">

</p>

<p>

<label for="isenha">Senha:</label> <input type="password" name="senha" id="isenha" required minlength="8" maxlength="16" size="14" placeholder="digite sua senha" autocomplete="current-password">

</p>

**Outros controles**

**Number**: para campos apenas com números, podendo limitar os números aceitos com **min** e **max=” ”** e permitindo também frações com **step=” ”**:

<p>

<label for="imedia">Nota média:</label>

<input type="number" name="media" id="imedia" placeholder="0 a 10" required min="0" max="10" step="0.1">

</p>

**Month, date** e **time** podem ser utilizados para campos de datas e horários:

<p>

            <label for="imes">Período Letivo</label>

            <input type="month" name="mes" id="imes" required>

        </p>

        <p>

            <label for="idia">Dia da Prova</label>

            <input type="date" required name="dia" id="idia">

        </p>

        <p>

            <label for="ihora">Horário da Prova</label>

            <input type="time" required name="hora" id="ihora">

        </p>

**Telefone e Email**

<p>

                <label for="iemail">Email:</label>

                <input type="email" name="email" id="iemail" required autocomplete="email" placeholder="Digite seu email">

            </p>

            <p>

                <label for="itel">Telefone:</label>

                <input type="tel" required placeholder="(xx) x xxxx-xxxx" name="tel" id="itel" autocomplete="tel">

            </p>

**Fieldset e título**

É possível agrupar os campos em grupos, utilizando **<fieldset>**, além de criar um título para cada grupo, utilizando **<legend>**:

<form action="cadastro.php" method="get" autocomplete="on">

        <fieldset>

            <legend>Dados pessoais</legend>

            <p>

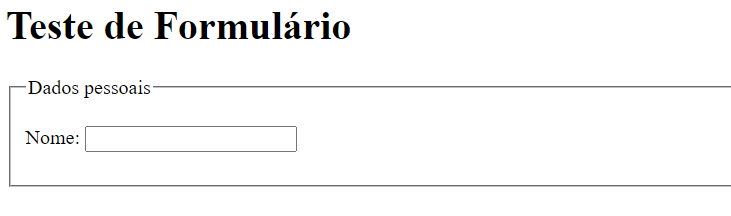
                <label for="inome">Nome:</label>

                <input type="text" name="nome" id="inome" required minlength="5" maxlength="20" autocomplete="name">

            </p>

        </fieldset>

    </form>



**Checkbox**

<p>

                <input type="checkbox" name="basq" id="ibasq"> <label for="ibasq">Basquete</label><br>

                <input type="checkbox" name="fut" id="ifut" checked> <label for="ifut">Futebol</label><br>

                <input type="checkbox" name="nat" id="inat"> <label for="inat">Natação</label><br>

                <input type="checkbox" name="cross" id="icross" checked> <label for="icross">Cross Fit</label><br>

            </p>

Utilizando “**checked**” a opção estará marcada automaticamente

**Radio button**

<p>

            <input type="radio" name="sexo" id="isxmas" value="M"> <label for="isxmas">Masculino</label>

            <input type="radio" name="sexo" id="isxfem" value="F"> <label for="isxfem">Feminino</label>

            </p>

Para corrigir o problema de não desmarcar e, ao mesmo tempo, permitir marcar apenas uma opção, manter o “**name**” igual em todas as opções, diferenciando apenas o “**id**”.

**Importante**: lembrar de sempre utilizar o “**value**” no inputs de radio.

**Elementos color, range e file em HTML**

**Color**: permite ao usuário que selecione uma cor. É possível configurar a cor mostrada inicialmente, utilizando **value** e informando o código hexadecimal da cor desejada:

<p>

<label for="icor">Cor:</label>

<input type="color" name="cor" id="icor" value="#1772E8">

</p>

**Range**: permite ao usuário que manuseie um medidor horizontal. É possível configurar o nível mínimo e o máximo permitido, utilizando **min** e **max** (o padrão é de 0 a 100). Também é possível configurar o nível inicial com **value** (o padrão será sempre na metade do medidor):

<p>

<label for="inivel">Nível de satisfação:</label>

<input type="range" name="nivel" id="inivel" min="0" max="10" value="7">

</p>

**File**: permite ao usuário que carregue e envie um arquivo. Ao trabalhar com arquivos lembrar de sempre utilizar o método **POST** ao invés de **GET**:

<p>

<label for="ifoto">Foto do perfil:</label>

<input type="file" name="foto" id="ifoto">

</p>

**Select, datalist e testarea em HTML**

**Select**: Permite ao usuário selecionar um item de uma lista pré-estabelecida. Criar um **option** para cada opção disponível. Caso queira criar grupos de opções, utilizar a <**optgroup>** e dentro dela **label=”nome-do-grupo”** para nomear:

<p>

<label for="iclube">Seu clube:</label>

<select name="clube" id="iclube" required>

<option value="NA" selected>--- Escolha ---</option>

<optgroup label="Nordeste">

<option value="BAH">Bahia</option>

<option value="CE">Ceará</option>

<option value="FOR">Fortaleza</option>

</optgroup>

<optgroup label="Sudeste">

<option value="COR">Corinthians</option>

<option value="FLA">Flamengo</option>

<option value="PAL">Palmeiras</option>

</optgroup>

</select>

</p>

**Datalist:** Assim como a **<select>**, permite ao usuário selecionar um item de uma lista pré-estabelecida, porém também possibilita-o de escrever alguma outra opção personalizada. Utilizar normalmente a **<input type=”text”>** + **list=”nome-da-lista”**. Para criar a lista, usar a <**datalist id=nome-da-lista>** e suas criar suas opções também com a **<option>**.

<p>

<label for="icargo">Cargo desejado:</label>

<input type="text" name="cargo" id="icargo" list="listacargo" required>

<datalist id="listacargo">

<option>Manager</option>

<option>Marketing</option>

<option>Comercial</option>

</datalist>

/p>

**Textarea:** cria um campo de texto para o usuário. Por padrão, o usuário pode redefinir o tamanho da caixa. Para evitar isso, utilizar o **style=”resize: none”**. Para limitar o número de caracteres digitáveis, utilizar a **maxlength:”x”**. Ao utilizar a **<textarea>** preferencialmente utilizar o método **POST** ao invés de **GET**:

<p>

<label for="imsg">Mensagem:</label> <br>

<textarea name="msg" id="imsg" cols="30" rows="10" style="resize:none" maxlength="100">

</textarea>

</p>

**Elemento Output**

Não envia dados, apenas mostra algum tipo de resultado na tela. Seguem 3 exemplos (**utilizando JavaScript**):

Ex1:

<p>

<label for="in1">Número 1: </label>

<input type="number" name="n1" id="in1" min="0" max="50" required oninput="isoma.innerHTML = Number(in1.value) + Number(in2.value)">

</p>

<p>

<label for="in2">Número 2: </label>

<input type="number" name="n2" id="in2" min="0" max="50" required oninput="isoma.innerHTML = Number(in1.value) + Number(in2.value)">

</p>

<p>

<label for="isoma">Soma: </label>

<output name="soma" id="isoma">0</output>

</p>

Ex2:

<p>

<label for="inum">Número: </label>

<input type="range" name="num" id="inum" min="0" max="10" value="5" oninput="ival.innerHTML = Number(inum.value)">

<output id="ival">5</output>

</p>

<p>

<input type="submit"> <input type="reset" value="Limpar">

</p>

Ex3:

<form action="cadastro.php" method="get">

        <p>

            <label for="iano">Em que ano você nasceu?</label>

            <input type="number" name="ano" id="iano" min="1923" max="2023" required oninput="calcIdade()">

        </p>

        <p>

            <label for="iidade">Sua idade é: </label>

            <output id="iidade">0</output>

        </p>

        <p>

            <input type="submit"> <input type="reset" value="Limpar">

        </p>

    </form>

    <script>

        function calcIdade() {

           let atual = new Date().getFullYear()

           iidade.innerHTML = Number(atual) - Number(iano.value)

        }

    </script>

**MEDIA QUERIES**

**Tela x Impressão**

Para criar diferentes versões para visualização em tela e impressão, pode-se criar duas CSS diferentes, especificando cada uma com o seu media type: **media=”screen”** e **media=”print”**

<link rel="stylesheet" href="estilos/tela.css" media="screen">

<link rel="stylesheet" href="estilos/impressora.css" media="print">

**Obs: em versão para impressão lembrar de priorizar fontes com serifa, monoespaçadas**

Media Query = Media Type + Media Feature

Media tipes -> tipos de mídia

Media features -> características de mídia

**Exibição**:

- As media features são exibidas dentro de ( ) e acompanham as media types dentro de " ", onde são unidas por 'and'.

<link rel="stylesheet" href="estilos/retrato.css" media="screen and (orientation:portrait)">

**Múltiplas Media Features:**

- Se tivermos mais de uma media feature, cada uma deve estar dentro de ( ) individualmente e unidas por 'and'.

<link rel="stylesheet" href="estilos/paisagem.css" media="screen and (min-width: 500px) and (orientation:landscape)">

**Reunindo tudo em um único CSS**

Pode ser feito usando a tag **<style>,** dentro de **<head>,** usando **@media** para especificar o tipo de mídia e **'and**' para unir os 'media features':

<style>

        /\* DECLARAÇÕES GERAIS \*/

        @media all {

            \* {

                padding: 0;

                margin: 0;

            }

        /\* DECLARAÇÕES RETRATO \*/

        @media screen and (orientation: portrait) {

            body {

                background-image: url(../imagens/cev-portrait.jpg);

                background-repeat: no-repeat;

                background-size: contain;

                background-position: center bottom;

            }

        }

Além disso pode ser feito também usando o 'link:css' para criar um arquivo de estilo separado com todas as configurações de todas as features.

**Typical Device Breakpoints**

<https://www.w3schools.com/howto/howto_css_media_query_breakpoints.asp>

**Pequenas telas**: até 600px

**Celular**: de 600px até 768px

**Tablet**: de 768px até 992px

**Desktop**: de 992px até 1200px

**Grandes telas**: acima de 1200px

Ex:

@media screen and (min-width: 768px) and (max-width: 992px) {

/\* TABLET \*/

}

@media screen and (min-width: 992px) and (max-width: 1200px) {

    /\* DESKTOP \*/

}

@media screen and (min-width: 1200px) {

    /\* GRANDES TELAS \*/

}

**Menu responsivo (hamburger)**

<https://fonts.google.com/icons> - ícone “**menu**”

**Web**

1. **Adicionar no <head>**:

    <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL,GRAD@20..48,100..700,0..1,-50..200" />

1. **Adicionar no local que deseja o menu:**

   <span class="material-symbols-outlined">menu</span>

1. **Configurar a CSS do ícone, inclusive o :hover**:

#burger {

    color: var(--cor1);

    background-color: var(--cor5);

    display: block;

    text-align: center;

    padding: 10px;

#burger:hover {

    background-color: var(--cor1);

    color: var(--cor5);

    cursor: pointer;

}

1. **Criar a condicional para expandir / retrair o menu com o clique:**

    </main>

    <script>

        function cliqueMenu() {

            if (idmenu.style.display == "block") {

                idmenu.style.display = "none"

            } else {

            idmenu.style.display = "block"

            }

        }

    </script>

</body>