**HTML 5**

**14/11 – HTML**

HTML é uma sigla para Hyper Text Markup Language, ou seja, Linguagem de Marcação de Hiper Texto (o Hiper Texto é como um link: ele é um texto que tem marcações e que permite você linkar (ou conectar) com outras referências). Portanto, NÃO é uma linguagem de programação, e sim de marcação. É composta por uma série de marcações que dizem para os servidores da web qual é o estilo e a estrutura de um documento. Com o HTML, os usuários podem criar e estruturar páginas da web, usando elementos, tags e atributos.

LINGUAGEM: conjunto de símbolos, palavras e regras utilizadas na construção de sentenças que expressam e processam informações para o computador.

MARCAÇÃO: determina a localização da informação

SEMANTICA: determina o sentido ou significado da informação.

Criar uma página em HTML é tão simples como formatar um documento no Word. Observe as semelhanças:



**TAGS**

Toda vez que uma marcação é feita em HTML, utilizamos **tags.** Essas tags são os comandos de marcação usados em HTML, elas são representadas assim:

INICIAR TAG: <marcação>

ENCERRAR TAG: </marcação>

Existem também tags que não possuem conteúdo, como: <marcação/>

Na tags, também há a existência de atributos: informações que são alocadas na tag de abertura que muda o comportamento dela.

<tag atributo=”valor”>texto</tag>

A estrutura de um arquivo HTML é a seguinte:

<!doctype html> (uma das únicas tags que não fecha, e indica para o computador que aquele arquivo é html)

<html></html> (é a tag mãe, ou seja, tag ancestral)

<head></head> e <body></body> (tags filhas da html)

Colocamos no <head> o título da página, configuração de idiomas, etc. No <body> se encontra o corpo do nosso site, é o que vai aparecer para o usuário.

**FERRAMENTAS PARA DESENVOLVEDOR**

No Google Chrome, existem ferramentas usadas por desenvolvedores para melhorar a visualização do código na página. Após pressionar F12 e Inspecionar Elemento, irá aparecer as seguintes colunas:

ELEMENTS: visualiza o código final que foi renderizado.

CONSOLE: executa o código de JavaScript diretamente na tela.

SOURCES: possui acesso aos recursos do site, tudo que esse site carrega.

NETWORK: monitora todas as requisições que o site realiza.

PERFORMANCE: mostra a performance de renderização da sua tela.

MEMORY: mostra o quanto de memória e processamento aquele site está requisitando.

APPLICATION: usada para gerenciar banco de dados, informações de sessão, cookies, dados locais, etc.

SECURITY: verifica se o site em questão é válido ou não em termos de segurança.

LIGHTHOUSE: mostra a performance do seu site, como boas práticas, acessibilidade, experiência do usuário. Enfim, é como uma auditoria.

**ESTRTURA BÁSICA DA PÁGINA**

A estrutura básica de uma página é a seguinte:

<!doctype html>

<html>

<head>

<meta charset=”UTF-8” // (permitir acentos e emojis)

<title>Titulo</title> // Título da página

</head>

<body>

<header>

<h1>Hcode</h1> // (Título em destaque)

<nav></nav> // (Navegação)

</header>

<section>

<div></div> // (Divisão)

</section>

<footer>

<p>Hcode Treinamentos</p> // (Pequeno texto)

</footer>

</body>

</html>

**15/11 – ATRIBUTOS GLOBAIS**

Como já aprendido, atributos são informações alocadas nas tags que mudam o comportamento delas. Porém, os atributos **globais** são atributos que podem ser alocados em qualquer tag. Eles são:

**accesskey:** define uma tecla de atalho para focar no elemento.

**class:** define quais classes CSS serão aplicadas no elemento.

**contenteditable:** define se o conteúdo do elemento pode ou não ser editável.

**data-\*:** usado para armazenar dados em um elemento. No lugar do (\*), deve-se definir um nome.

**dir:** define a direção do conteúdo de um elemento.

**draggable:** define se um elemento pode ser arrastado ou não.

**dropzone:** define o que acontece quando o elemento arrastado é solto neste elemento.

**hidden:** define o elemento como oculto.

**id:** define uma identificação única para o elemento.

**lang:** define a linguagem do conteúdo do elemento.

**spellcheck:** define se o conteúdo do elemento deve ter sua ortografia verificada.

**style:** define estilos CSS diretamente no elemento.

**tabindex:** define a posição do elemento na ordem da tabulação da página.

**title:** define informações sobre o elemento.

**translate:** define se o conteúdo do elemento pode ser traduzido ou não.

**EXEMPLO DE USO DOS ATRIBUTOS GLOBAIS**

**accesskey:** define uma tecla de atalho para focar no elemento.

<body>

    <button type="button" accesskey="n">Novo</button>

    <button type="button" accesskey="s">Salvar</button>

</body>

No código acima, eu crio um botão usando a tag **button** e defino uma accesskey para ele, ou seja, uma tecla de atalho que, se pressionada, focará nele. Nesse caso, para o botão “Novo”, eu usei N e para o botão “Salvar”, usei S.

--------------------------------------------------------------------------------------------------------------------

**class:** define quais classes CSS serão aplicadas no elemento.

<head>

    <style>

        .botao{

            border: none;

            background-color: bisque;

            border-radius: 5px;

            padding: 10px 25px;

            font-family: Arial, Helvetica, sans-serif;

            font-size: 12px;

            text-decoration: none;

        }

    </style>

</head>

<body>

    <button type="button" class="botao">OK</button>

    <a href="a" class="botao">Link</a>

No código acima, criei dois botões: OK e Link. Para os dois, eu defini que usariam a mesma classe de CSS usando o atributo <class>. E um pouco antes, dentro da tag <head>, eu usei uma tag chamada <style> e defini o estilo daquela classe (tamanho da borda, fonte, tamanho da fonte, etc).

--------------------------------------------------------------------------------------------------------------------

**contenteditable:** define se o conteúdo do elemento pode ou não ser editável.

<body>

    <p contenteditable="true">Lorem, ipsum dolor sit amet consectetur adipisicing elit. Pariatur, fugiat, eos laudantium labore ut molestiae deleniti nemo dolorem aperiam nisi voluptatem eveniet dolorum ipsam autem cupiditate libero consequatur. Fugiat, laborum?</p>

</body>

Nesse código, criei um parágrafo com uma frase em latim usando a tag<p>, e usei o atributo contenteditable nela (+ true para indicar que a condição de texto editável é verdadeira) para que o usuário possa editar essa frase. Isso não modifica o arquivo, apenas o site na visão do usuário.

----------------------------------------------------------------------------------------------------------

**data-\***: usado para armazenar dados em um elemento. No lugar do (\*), deve-se definir um nome.

<body>

    <ul>

        <li data-id="1" data-email="joaorangel@gmail.com">João Rangel</li>

        <li data-id="2" data-email="glaucio@hcode.com.br">Glaucio Daniel</li>

    </ul>

</body>

Nesse código, criei uma lista com dois elementos: João Rangel e Glaucio Daniel. Porém, salvei dois tipos de dados nesses elementos usando o atributo data: o id e o email. Eles estão ocultos para o usuário, mas estão salvos no arquivo. Eu posso puxar esses dados para a visibilidade do usuário no console por meio de JavaScript.

--------------------------------------------------------------------------------------------------------------------

**dir:** define a direção do conteúdo de um elemento.

<body>

    <p dir="ltr">Lorem ipsum dolor sit amet consectetur, adipisicing elit. Placeat, similique, ipsa nisi ullam officia facilis doloremque earum sint corrupti numquam quisquam iusto voluptate esse! Illo asperiores vero at quis repudiandae.</p>

    <p dir="rtl">Lorem ipsum dolor sit amet consectetur adipisicing elit. Pariatur aliquam ratione id eius aut ducimus eligendi totam eum, asperiores fugiat aspernatur hic architecto. Sit suscipit tempore assumenda esse, a quas.</p>

</body>

No código acima, eu criei dois textos usando a tag <p>. Porém, no primeiro texto, usando o atributo dir, eu defini que a direção de exibição desse texto seria da esquerda para a direita (left to right). Já no segundo texto, a direção foi definida como sendo da direita para esquerda (right to left). Esse atributo é muito útil quando o site é exibido em idiomas que possuem essa característica (árabe, hebraico, etc).

--------------------------------------------------------------------------------------------------------------------

**draggable:** define se um elemento pode ser arrastado ou não.

<head>

    <style>

        #quadrado{

            width: 100px;

            height: 100px;

            background-color: greenyellow;

        }

    </style>

</head>

<body>

    <button type="button" draggable="true">OK</button>

    <div id="quadrado" draggable="true"></div>

</body>

No código acima, eu criei um botão de OK usando a tag <button> e um quadrado dentro de uma <div>. Dei uma id para o quadrado para usar a tag <style> nele, deixando-o visível e mais bonito. Tanto no quadrado como no botão, usei o atributo draggable para tornar possível que o usuário arraste esses objetos dentro do site.

--------------------------------------------------------------------------------------------------------------------

**dropzone:** define o que acontece quando o elemento arrastado é solto neste elemento.

<head>

    <style>

        .caixa{

            width: 100px;

            height: 100px;

            background-color: orange;

        }

        .drop{

            width: 200px;

            height: 200px;

            border: 1px crimson solid;

        }

    </style>

</head>

<body>

    <div class="caixa" draggable="true"></div>

    <div class="drop" dropzone="copy"></div>

</body>

No código acima, criei dois quadrados usando a tag <div>. Defini o tamanho, cor e borda desses quadrados usando CSS por meio da tag <style>. Um representa uma caixa que pode ser arrastada (declarei essa condição usando o atributo draggable). O outro quadrado é bem maior e representa a zona onde vou dropar a caixa. Usei o atributo dropzone nesse quadrado para definir que é nele em que a caixa pode ser dropada.

--------------------------------------------------------------------------------------------------------------------

**hidden:** define o elemento como oculto.

<body>

    <div>Hcode</div>

    <div hidden>Treinamentos</div>

</body>

No código acima, criei dois textos usando a tag <div>. O texto “Hcode” está normal, mas o texto “Treinamentos” está oculto, justamente por causa do uso do atributo hidden. A palavra “Treinamentos” aparece no código, mas não aparece para o usuário.

--------------------------------------------------------------------------------------------------------------------

**id:** define uma identificação única para o elemento.

<head>

    <style>

        #box-1{

            width: 150px;

            height: 150px;

            background-color: blueviolet;

        }

        #box-2{

            width: 150px;

            height: 150px;

            background-color: aqua;

        }

    </style>

</head>

<body>

    <div id="box-1"></div>

    <div id="box-2"></div>

</body>

No código acima, criei duas caixas usando a tag <div>. Dei um id para elas e agora elas são identificadas de acordo com seus nomes “box-1” e “box-2”. Usando essa identificação, criei um estilo de CSS para elas usando a tag <style>.

--------------------------------------------------------------------------------------------------------------------

**lang:** define a linguagem do conteúdo do elemento.

<body>

    <h1 lang="pt-br">Olá mundo!</h1>

    <h2 lang="en">Hello World!</h2>

    <h3 lang="fr">Bonjour le monde!</h3>

</body>

Nesse código, criei três textos usando as tags <h1>, <h2> e <h3>, porém com idiomas diferentes. Defini o idioma desses textos antes por meio do atributo lang e a respectiva sigla do idioma correspondente. Visualmente, não muda nada, mas essa definição é importante para os motores de busca e para uma possível tradução da página.

--------------------------------------------------------------------------------------------------------------------

**spellcheck:** define se o conteúdo do elemento deve ter sua ortografia verificada.

<body>

    <p spellcheck="true" contenteditable="true">Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quos amet, vero, a iste incidunt unde aspernatur expedita ex hic consequuntur nobis dignissimos iure deserunt odio quia aliquam quasi illum ducimus.</p>

    <p spellcheck="false" contenteditabler="true">Lorem ipsum dolor, sit amet consectetur adipisicing elit. Possimus quisquam ullam, tenetur incidunt dolore laudantium pariatur eum veritatis fugit quam consequatur fuga id ab, officiis, commodi et sequi! Iusto, ipsum!</p>

</body>

No código acima, criei dois textos usando a tag <p>. Porém, no primeiro texto, usei o atributo spellcheck, para que o navegador verifique a ortografia daquele texto. Foi necessário também usar o atributo contenteditable, pois o navegador só corrige a ortografia de um texto que o usuário coloca.

--------------------------------------------------------------------------------------------------------------------

**style:** define estilos CSS diretamente no elemento.

<body>

    <p style="color:lime">Lorem ipsum dolor sit amet consectetur adipisicing elit. Nam dicta quia repellendus! Atque eos dolores quibusdam incidunt saepe aliquam, alias placeat eius iure nisi est ipsam, temporibus delectus veritatis laboriosam!</p>

</body>

Nesse código, criei um texto usando a tag <p>. Porém, defini uma cor para esse texto usando o atributo style. É possível também definir outras características, como fonte, tamanho do texto, etc. É possível também definir o style separadamente para um id ou class por meio da própria tag <style>.

--------------------------------------------------------------------------------------------------------------------

**tabindex:** define a posição do elemento na ordem da tabulação da página.

<body>

    <form>

        <input type="text" placeholder="E-mail" tabindex="1">

        <input type="password" placeholder="Senha" tabindex="2">

        <button type="reset" tabindex="4">Limpar</button>

        <button type="submit" tabindex="3">Enviar</button>

    </form>

</body>

Nesse código, criei um formulário com dois campos de texto (email e senha) e dois botões (Limpar e Enviar). Defini por meio do atributo tabindex que, ao apertar tab pela primeira vez, a caixa selecionada é “E-mail”. Ao apertar tab novamente, irá para “Senha”. Apertando tab de novo, irá para “Enviar” e o tab pressionado pela quarta vez selecionará o botão “Limpar”.

----------------------------------------------------------------------------------------------------------

**title:** define informações sobre o elemento.

<body>

    <a href="https://youtube.com/HcodeBrasil" title="Canal da Hcode no Youtube">Hcode</a>

</body>

No código acima, criei uma palavra (Hcode) que possui um hiperlink para o YouTube. Usei o atributo title para que, quando o usuário passar o mouse em cima do nome Hcode, ele veja do que se trata aquele link (Canal da Hcode no Youtube),

----------------------------------------------------------------------------------------------------------

**translate:** define se o conteúdo do elemento pode ser traduzido ou não.

<body>

    <h1 translate="no">Hello World!</h1>

    <h2 translate="yes">Welcome!</h2>

</body>

No código acima, criei duas frases em inglês, porém usei o atributo translate para determinar que o Hello World não fosse traduzido.

**16/11 – TAGS (PARTE 1)**

**<html>**: tag principal, usada para definir um documento HTML. Todo o conteúdo do nosso documento deve estar dentro dessa tag.

**<!DOCTYPE html>**: única tag que pode vir antes da <html>, usada para que o navegador reconheça que aquele arquivo é um HTML5.

**<!-- Comentário -->**: é algo que não fica visível para o usuário, mas permanece no código para que o desenvolvedor se localize.

**<body>**: é a tag onde se localiza o corpo do nosso documento HTML, ou seja, todos os elementos de layout, os elementos visíveis da nossa página. É filha da tag <html>.

**<head>**: é a tag onde configuramos o cabeçalho do documento HTML. Nessa tag, pode ser incluído o titulo do documento, estilos, scripts, meta informações, entre outros. É filha da tag <html>.

**<title>**: tag usada para definir o titulo do documento. Esse título sempre aparece na janela do navegador e também quando o usuário adiciona a página aos favoritos. É filha da tag <head>.

**<meta>**: usada para definir um metadado da página. Esses metadados são usados pelos navegadores e mecanismos de pesquisa e outros. Essa tag NÃO precisa ser fechada. É filha da tag <head>. Possui alguns atributos específicos, que são:

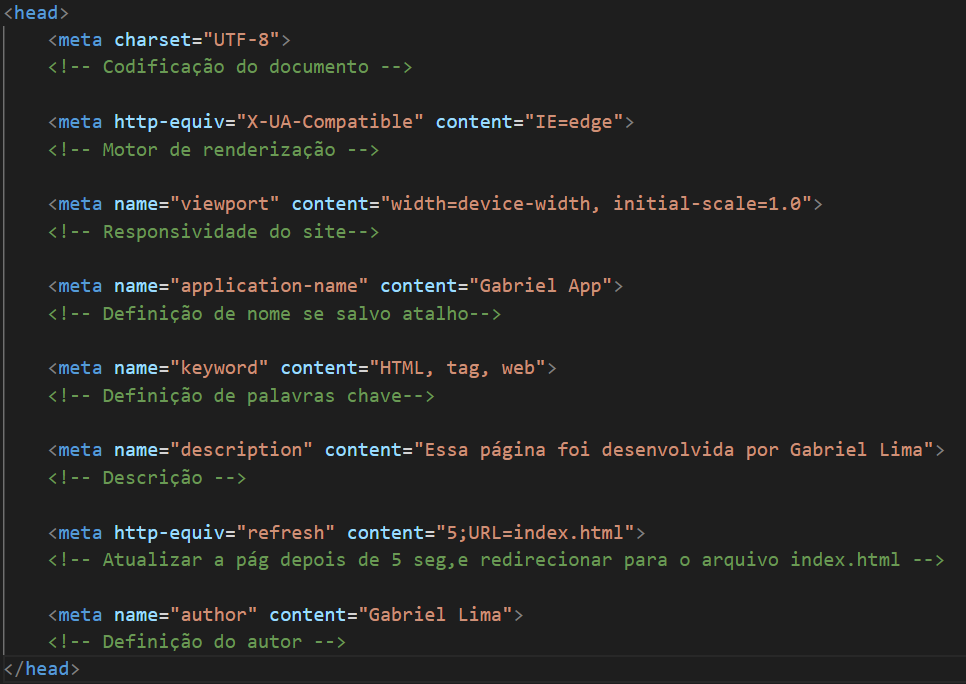
*charset* = usado para definir a codificação de caracteres usada no documento (permitir acentos)

*content* = define o conteúdo associado ao http-equiv.

*http-equiv* = provê um cabeçalho HTTP com informações do conteúdo do atributo.

*name* = define o nome do metadado.

Observe um exemplo do uso da tag <meta>:



**<p>**: tag usada para definir um parágrafo. É filha da tag <body>.

**<h1>**: tag usada para definir o titulo principal do documento HTML, o primeiro cabeçalho mais importante da página. É filha da tag <body>.

**<h2>:** tag usada para definir o segundo cabeçalho mais importante da página. É filha da tag <body>.

**<h3>:** tag usada para definir o terceiro cabeçalho mais importante da página. É filha da tag <body>.

**<h4>:** tag usada para definir o quarto cabeçalho mais importante da página. É filha da tag <body>

**<h5>:** tag usada para definir o quinto cabeçalho mais importante da página. É filha da tag <body>.

**<h6>:** tag usada para definir o sexto cabeçalho mais importante da página. É filha da tag <body>.

**<style>:** tag usada para definir um estilo em CSS. Possui dois atributos próprios:

*media* = define qual dispositivo de mídia o estilo se aplica.

*type* = define o tipo de mídia.

Observe um exemplo do uso da tag <style>:

<head>

    <style>

        body{

            background-color: darkkhaki;

        }

    </style>

    <!-- Definindo a cor do plano de fundo do site  -->

    <style media="print">

        body{

            background-color:aqua;

        }

    </style>

    <!-- Definindo a cor do plano de fundo

    caso o usuário imprima o conteúdo do site-->

</head>

<body></body>

**17/11 – TAGS (PARTE 2)**

**<script>**: tag usada para definir um script em JavaScript. A tag permite o script de ser escrito dentro dela ou apontar para um fonte externa (outro arquivo de formato .js). Essa tag pode ser usada tanto no <head> como no <body>, porém é mais recomendado que ela seja usada dentro do <body>, um pouco antes dessa tag fechar. Isso porque, se ela está na tag <head>, o navegador interrompe a renderização, trazendo lentidão ao carregamento da página e isso não é uma boa prática. Caso a tag <script> seja colocada em <head>, pode acontecer um bloqueio de renderização do site.

Possui cinco atributos próprios:

*async* = define que o script será executado de forma assíncrona (funciona apenas para scripts externos).

*charset* = define o padrão de caracteres usado.

*defer* = define que o script só será executado depois da página ser carregada.

*src* = define a fonte externa do script.

*type* = define o tipo de mídia do script.

**<div>:** tag usada para definir uma divisão ou seção no documento. Muitas vezes, o principal objetivo da tag <div> é estilizar com CSS ou executar tarefas com JavaScript. É muito usada na tag <body>.

**<span>:** tag usada para agrupar textos ou tags no documento. Não fornece nenhuma alteração visual por padrão. Normalmente, o objetivo de seu uso é aplicar algum estilo CSS. É muito usada na tag <body>.

**<link>:** tag que NÃO tem fechamento, usada para definir uma associação entre um documento atual e um externo. O uso mais comum é para associar um documento CSS. É muito usada na tag <head>. Possui sete atributos próprios:

*crossorigin* = define como o elemento lida com solicitações de origem cruzada.

*href* = define o link referente ao documento que será associado.

*hreflang* = define a linguagem do documento associado.

*media* = define em qual dispositivo se aplica esse documento.

*rel* = define o relacionamento entre o documento atual e o associado.

*sizes* = define um tamanho de largura e altura para a associação de ícones.

*type* = define o tipo de mídia do documento associado.

Observe o uso dos relacionamentos na tag <link>:

**

**<noscript>:** tag usada para definir um conteúdo alternativo para usuários que desativaram scripts no navegador. O conteúdo dentro dessa tag só será exibido se os scripts não forem suportados. Pode ser usada tanto em <head> como em <body>.

**<embed>:** tag usada para definir um conteúdo externo embutido (como flash, por exemplo). Possui quatro atributos próprios:

*height* = define a altura do conteúdo embutido.

*src* = define a fonte do conteúdo embutido.

*type* = define o tipo do conteúdo embutido.

*width* = define a largura do conteúdo embutido.

**<object>:** tag usada para definir um objeto incorporado no documento. É possível incorporar arquivos de áudio, vídeo, PDF, etc. É usado na tag <body>. Possui sete atributos próprios:

*data* = define a URL do recurso que será usado como objeto.

*form* = define a que formulário esse objeto pertence.

*height* = define a altura do objeto.

*name* = define o nome do objeto.

*type* = define o tipo de mídia do objeto.

*usemap* = define o nome do mapa de imagem para ser usado com o objeto.

*width* = define a largura do objeto.

**<param>**: essa tag NÃO tem fechamento. Ela é usada para definir um parâmetro para plugins incorporados com a tag <object>. Possui dois atributos próprios:

*name* = define o nome do parâmetro.

*value* = define o valor do parâmetro.

**<base>:** essa tag NÃO tem fechamento. Usada para definir um URL padrão para todos os links da página. Pode haver no máximo uma tag <base> dentro da tag <head>. Possui dois atributos próprios:

*href* = define o URL base para todos os links da página.

*target* = define onde o link deverá ser aberto.

**<iframe>:** essa tag é usada para definir um quadro embutido. O quadro embutido é usado para incorporar outro documento no documento HTML atual. Possui seis atributos próprios:

*height* = define a altura do quadro.

*name* = define o nome do quadro.

*sandbox* = ativa conteúdos restritos por padrão.

*src* = define qual a fonte do documento a ser embutido.

*srcdoc* = define qual a estrutura do documento a ser embutido.

*width* = define a largura do quadro.

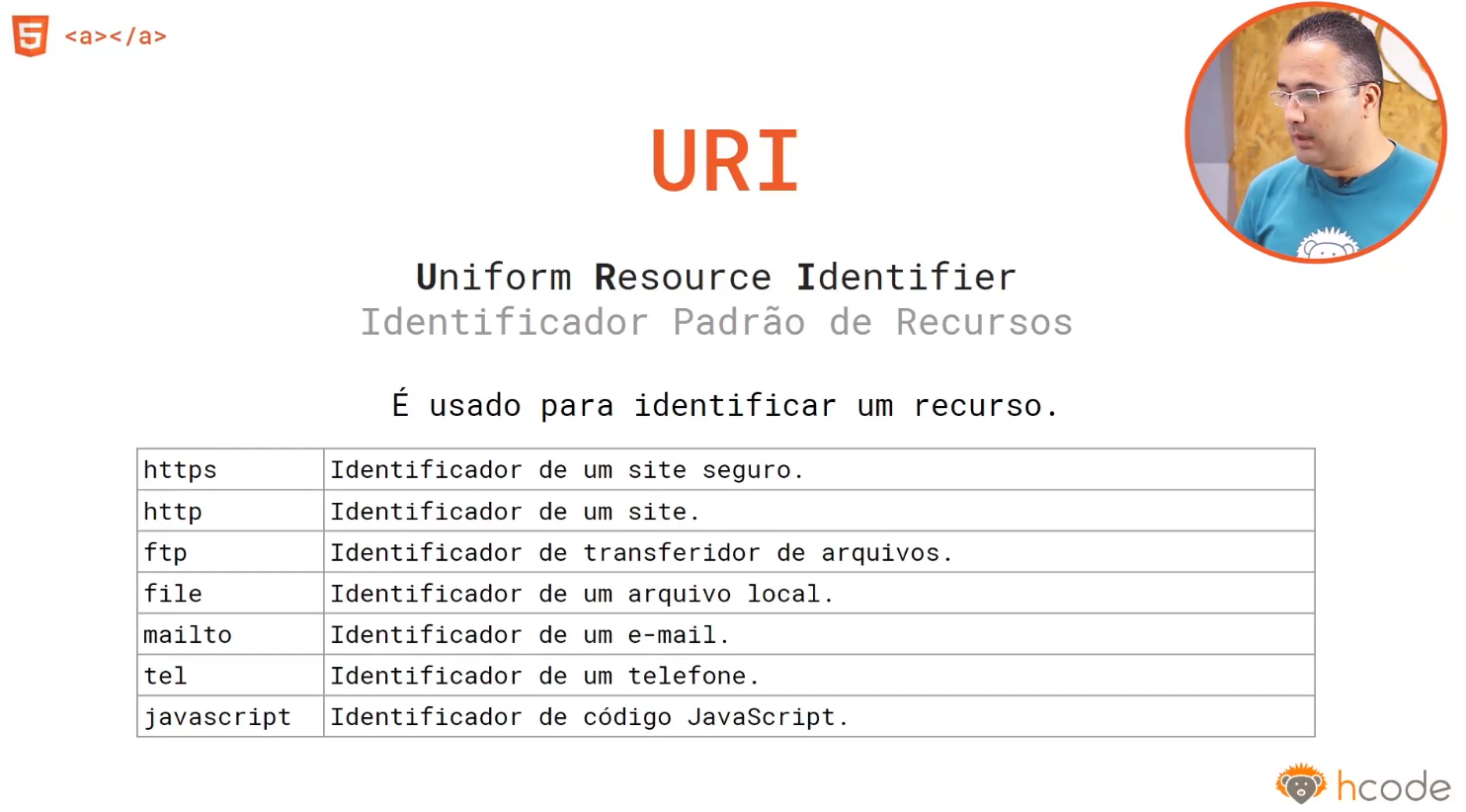
**18/11 – TAGS (PARTE 3)**

**<a>:** tag usada para criar um hyperlink ou vincular uma página HTML com outra. O atributo mais utilizado nessa tag é o href.

Na tag <a>, faremos muito uso de URL, que significar “Uniform Resource Locator” (Localizador Padrão de Recursos). É o endereço de um recurso disponível em uma rede local ou remota. Essa é a estrutura de um URL:



Dentro de uma URL, na parte do protocolo, temos a URI: “Uniform Resource Identifier” (Identificador Padrão de Recursos), que são eles:



Por padrão, o conteúdo dessa tag é mostrado sublinhado. Possui nove atributos próprios:

*download* = usado para forçar o download do destino do link.

*href* = define qual o endereço do link de destino.

*hreflang* = define qual a linguagem do link de destino.

*media* = define qual a mídia ou dispositivo do link de destino.

*ping* = endereço de rastreio para monitorar cliques no link de destino.

*referrerpolicy* = define qual referenciador enviar.

*rel* = define qual a relação do documento atual e o de destino.

*target* = define onde o link deverá ser aberto. ( *\_self* = na mesma aba; *\_blank* = em outra aba)

*type* = define o tipo de mídia do link de destino.

**<img>:** essa tag NÃO tem fechamento. É uma tag usada para inserir uma imagem na página. Possui 10 atributos específicos (os atributos src e alt são obrigatórios):

*alt* = define um texto alternativo para a imagem.

*crossorigin* = define como o elemento lida com solicitações de origem cruzada.

*height* = define a altura da imagem.

*ismap* = define a imagem como um mapa de imagem do lado do servidor.

*longdesc* = define uma descrição detalhada da imagem.

*sizes* = define tamanhos alternativos da imagem.

*src* = define qual a fonte/link da imagem.

*srcset* = define a URL da imagem em diferentes situações.

*usemap* = define a imagem como um mapa de imagem ao lado do cliente.

*width* = define a largura da imagem.

**<figure>:** tag usada para definir uma foto, ilustração, diagrama, etc.

**<figcaption>:** tag que fica dentro da tag <figure>, usada para definir uma legenda para a essa tag.

**<audio>:** tag usada para definir um som, música ou transmissões de áudio. Atualmente suporta os formatos mp3, wav e ogg. Qualquer texto no conteúdo da tag será exibido em navegadores que não suportam a tag <audio>. Possui 6 atributos próprios:

*autoplay* = começa a reproduzir assim que os dados do áudio estiverem carregados.

*controls* = habilita os controles de reprodução do áudio.

*loop* = define se o áudio deve começar a reproduzir novamente após terminar.

*muted* = define o volume do áudio como mutado.

*preload* = define como o áudio deve ser carregado quando a página é carregada.

*src* = define qual a fonte/link do áudio.

**<vídeo>:** tag usada para definir um vídeo, filme ou transmissão de vídeo. Atualmente, suporta os formatos mp4, webm e ogg. Qualquer texto no conteúdo da tag será exibido em navegadores que não suportam a tag <video>. Possui 9 atributos específicos:

*autoplay* = começa a reproduzir assim que os dados do vídeo estiverem carregados.

*controls* = habilita os controles de reprodução do vídeo.

*height* = define a altura do player do vídeo.

*loop* = define se o vídeo deve começar a reproduzir novamente após terminar.

*muted* = define o volume do vídeo como mutado.

*poster* = define uma imagem para ser exibida enquanto o vídeo estiver sendo carregado ou antes do usuário clicar em reproduzir.

*preload* = define como o vídeo deve ser carregado quando a página é carregada.

*src* = define qual a fonte/link do vídeo.

*width* = define a largura do player do vídeo.

**<source>:** essa tag NÃO tem fechamento. Tag usada para definir uma fonte de mídia. É usada junto com as tags <audio>, <video> e <picture>. Possui cinco atributos próprios:

*src* = define a fonte externa do source.

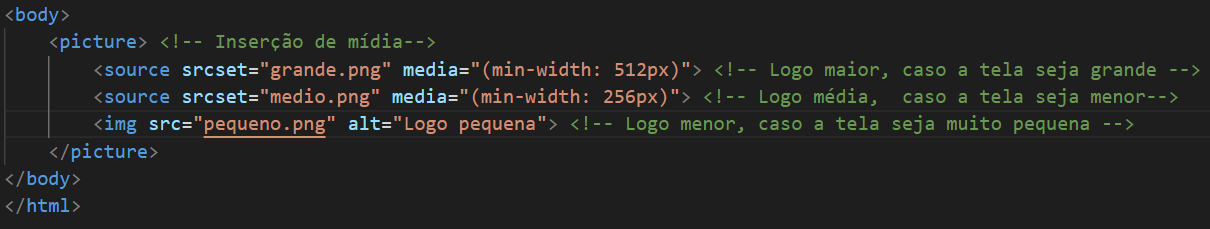
*srcset* = define uma fonte em situações diferentes.

*media* = define em que tipo de mídia essa fonte de dados se aplica.

*sizes* = define tamanhos diferentes.

*type* = define o tipo de mídia do arquivo.

**<picture>:** tag usada para definir mais de uma fonte de imagem. Normalmente usada para mostrar imagens de acordo com o tamanho da tela. Essa tag deve ter uma tag <img> e uma ou mais tags <source>. Então a ordem é:

**

**22/11 – TAGS (PARTE 4)**

**<track>:** essa tag NÃO tem fechamento. É usada para definir uma faixa de texto em uma tag de mídia. Normalmente usada para definir legendas em vídeos. Possui cinco atributos próprios:

*default* = define se esta é a faixa selecionada por padrão.

*kind* = define o tipo de faixa de texto.

*label* = define o rótulo da faixa de texto.

*src* = define a URL da fonte da faixa de texto.

*srclang* = define qual a linguagem dos dados da faixa de texto. Obrigatório código de linguagem para legendas.

**<table>:** tag usada para definir uma tabela. Porém, apenas a tag table não cria toda a tabela. É necessário usar as tags <tr> e <td> para criar linhas e células respectivamente. Uma tabela mais complexa ainda pode conter as tags <caption>, <col>, <colgroup>, <thead>, <tfoot> e <tbody>. Historicamente, as tabelas eram usadas para criar o layout da página, porém esse seria um mau uso dessa tag.

**<tr>:** (table row) Tag usada para definir uma linha dentro de uma tabela. Esta tag deve conter um ou mais tags <th> e/ou <td>.

**<td>**: (table) Tag usada para definir uma célula da tabela. Deve ser usada dentro de uma tag <tr>. Possui três atributos próprios:

*colspan* = define o número de colunas que esta célula deve ocupar.

*headers* = define uma ou mais células de cabeçalho relacionadas.

*rowspan* = define o número de linhas que essa célula deve ocupar.

**<thead>:** tag usada para agrupar o conteúdo do **cabeçalho** da tabela. É usada em conjunto com as tags <tbody> e <tfoot>. Os navegadores podem usar essa definição para ativar a rolagem do corpo da tabela. Essa tag deve ser filha de uma tag <table>, vir antes das tags <tbody> e <tfoot> e ter uma ou mais tags <tr> dentro dela.

**<tbody>:** tag usada para agrupar o **corpo** de uma tabela. É usada em conjunto com as tags <thead> e <tfoot>. Os navegadores podem usar essa definição para ativar a rolagem do corpo da tabela. Essa tag deve ser filha de uma tag <table> e deve vir depois da tag <thead> e antes da tag <tfoot>.

**<tfoot>:** tag usada para agrupar o rodapé de uma tabela. É usada em conjunto com as tags <thead> e <tbody>. Os navegadores podem usar essa definição para ativar a rolagem do corpo da tabela. Essa tag deve ser filha de uma tag <table>, deve vir depois das tags <thead> e <tbody> e deve ter uma ou mais tags <tr> dentro dela.

**<th>:** tag usada para definir uma célula de cabeçalho. Deve ser usada dentro de uma tag <tr>. Substitui a tag <td> dentro do <thead> numa tabela. Possui seis atributos próprios:

*abbr* = define uma abreviação do conteúdo de uma célula do cabeçalho.

*colspan* = define o número de colunas que esta célula deve ocupar.

*headers* = define uma ou mais células de cabeçalho relacionadas.

*rowspan* = define o número de linhas que essa célula deve ocupar.

*scope* = define se a célula é um cabeçalho de linha, coluna, grupo de linha ou grupo de coluna.

*sorted* = define a direção da ordem da coluna.

**<template>:** tag usada para definir um contêiner para código HTML que deseja renderizar posteriormente.

**<colgroup>:** tag usada para definir propriedades em todas as células que formam uma coluna. Deve vir dentro da tag <table>. Possui apenas um atributo próprio:

*span* = define o número de colunas que esse grupo deve abranger.

**<col>:** tag que NÃO tem fechamento. Define propriedades em todas as células que formam uma coluna, mas diferente da tag colgroup que define as propriedades de uma forma geral, a tag <col> define de forma personalizada, para cada coluna. Possui apenas um atributo próprio:

*span* = define o número de colunas que esse grupo deve abranger.

**<caption>:** tag usada para definir uma legenda da tabela. Deve ser inserida imediatamente após a abertura da tag <table>.

**23/11 – TAGS (PARTE 5)**

**<header>:** tag usada para definir o cabeçalho de uma página. Normalmente essa tag possui tags de título como h1, h2, h3, h4, h5 e h6. Pode ser colocado um logotipo, ícone ou autoria em seu conteúdo.

**<main>:** tag usada para definir o conteúdo principal e único do elemento. Só deve haver uma tag <main> por página, e não deve haver nenhum conteúdo repetido nessa tag, como barras laterais, links de navegação, logotipos e etc. Visualmente falando, tanto essa como as próximas tags não mudam em nada, mas elas tem um peso semântico, que vai contribuir com os motores de busca e com o SEO do site.

**<footer>:** tag usada para definir o rodapé de uma página. Essa tag normalmente contém informações sobre direitos autorais, mapa do site, link para o topo, links para outras páginas, etc.

**<article>:** tag usada para definir como artigo o conteúdo da página. Normalmente é usada em fóruns, blogs e portais de notícias. É importante para separar semanticamente o conteúdo.

**<section>:** tag usada para definir uma seção no documento. Antes de usar essa tag, é preciso analisar que seção será definida. Se não for um cabeçalho, rodapé, nem parte principal, aí sim é bom usar a tag <section>.

**<aside>:** a tag define um conteúdo ao lado do artigo (na lateral da página). É importante para separar semanticamente o conteúdo.

**<nav>:** essa tag define um conjunto de links de navegação. Visualmente não muda em nada, mas possui um peso semântico.

**<address>:** tag usada para definir as informações de contato do autor do documento. Se estiver dentro da tag <article>, irá definir as informações do autor do artigo. A tag pode incluir o endereço físico, URL, e-mail, telefone, mídias sociais, coordenadas geográficas, etc. Deve incluir o nome da pessoa, grupo ou organização.

**<details>:** tag usada para definir detalhes adicionais que o usuário pode exibir ou ocultar. Qualquer tipo de conteúdo pode ser colocado dentro dessa tag. A tag deve ter uma tag <summary> dentro dela. Possui um atributo próprio:

*open* = define se os detalhes devem ficar visíveis ao usuário.

**<summary>:** tag usada para definir o cabeçalho visível da tag <details>. Deve estar sempre dentro dessa tag.

**<ul>:** tag usada para definir uma lista não ordenada.

**<ol>:** tag usada para definir uma lista ordenada. A ordem da lista pode ser alfabética ou numérica. Possui três atributos próprios:

*reversed* = define a ordem da lista como decrescente.

*start* = define qual o valor de início da lista.

*type* = define o tipo de marcação de ordem da lista.

**<li>:** tag usada para definir um item da lista. Possui um atributo próprio:

*value* = define o valor do item da lista. Quando usada em listas ordenadas, esse será o valor inicial da lista.

**24/11 – TAGS (PARTE 6)**

**<map>:** tag usada para criar um mapa de imagem clicável. Essa tag define as áreas clicáveis usando várias tags <area>. Possui um atributo próprio (que é obrigatório):

*name* = define um nome para relacionar com o atributo usemap da tag img.

**<area>:** essa tag NÃO tem fechamento. É usada para definir áreas clicáveis em um mapa de imagem. Deve sempre vir dentro de uma tag <map>. Possui vários atributos próprios:

*alt* = define um texto alternativo para a imagem.

*coords* = define as coordenadas da área.

*download* = usado para forçar o download do destino do link.

*href* = define qual o endereço do link de destino.

*hreflang* = define qual a linguagem do link de destino.

*media* = define qual a mídia ou dispositivo do link de destino.

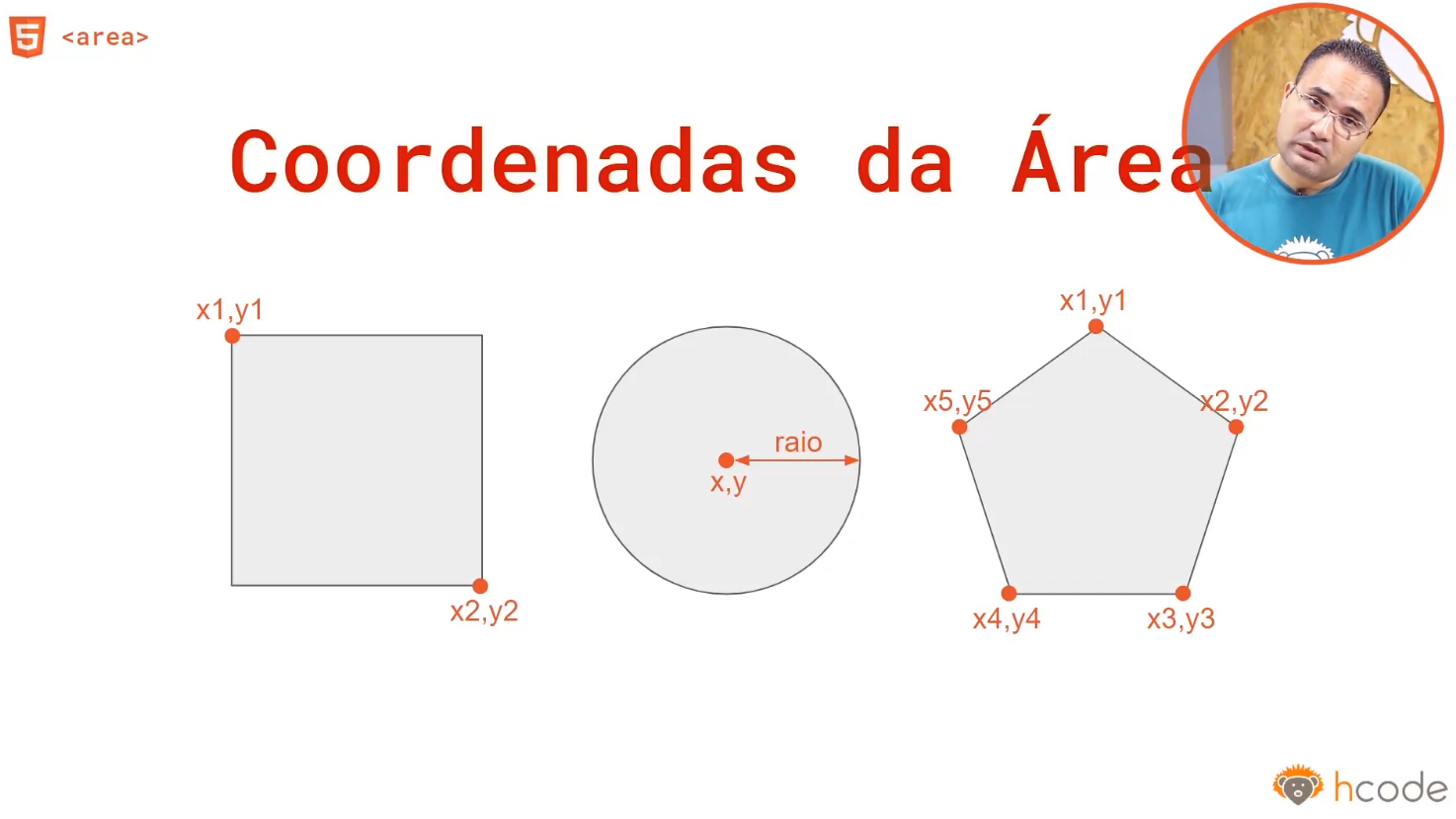
*rel* = define qual a relação do documento atual e o de destino.

*shape* = define uma forma para a área.

*target* = define onde o link deverá ser aberto.

*type* = define o tipo de mídia do link de destino.

Para determinar as coordenadas da tag <area>, fazemos assim:

**

x1, y1, x2, y2 para shape rect

x1, y1, raio para shape circle

x1, y1, x2, y2, x3, y3… para shape poly

**<abbr>:** usada para definir uma abreviação. É importante para fornecer informações úteis para navegadores, sistemas de tradução e mecanismos de busca (possui peso semântico).

**<b>:** tag usada para definir um texto em negrito, mas sem nenhum peso semântico. Antes de usar essa tag, é importante considerar o uso de outras mais semânticas, como <h1> a <h6>, <mark>, <strong> e <em>.

**<bdi>:** tag usada para definir parte de um texto que pode ser formatado em uma direção diferente do padrão. Essa tag é útil ao exibir conteúdo gerado pelo usuário com uma direção desconhecida.

**<bdo>:** tag usada para definir a direção do texto. Essa tag substitui a direção do texto atual. Possui o seguinte atributo:

*dir* = define a direção do texto dentro do elemento.

**<blockquote>:** tag usada para definir uma citação externa. Para citações curtas, é melhor usar a tag <q>. Possui um atributo próprio:

*cite =* define a URL da fonte da citação.

**<q>:** tag usada para definir uma situação simples. Possui um atributo próprio:

*cite =* define a URL da fonte da citação.

**26/11 – TAGS (PARTE 7)**

**<br>:** tag que NÃO possui fechamento, usada para definir uma quebra de linha. É importante usar essa tag para quebrar linhas, e não para separar parágrafos. Em caso de separação de parágrafos, deve-se usar duas tags <p>. Não é recomendado usar essa tag para empurrar algo para baixo. Ao invés disso, deve-se usar CSS.

**<wbr>:** tag que NÃO possui fechamento, usada para definir onde no texto é aceitável adicionar uma quebra de linha se necessário.

**<cite>:** tag usada para definir o título de uma obra (livro, música, filme, pintura, escultura, etc).

**<data>:** tag usada para associar um texto com algum outro dado. Ela é importante para deixar o conteúdo legível tanto para humanos quanto para programas. Usa-se o atributo value.

**<del>:** tag usada para definir um texto que foi excluído. Possui dois atributos próprios:

*cite* = define a fonte do motivo pelo qual o texto foi excluído.

*datetime* = define quando o texto foi excluído,

**<ins>:** tag usada para definir um texto que foi inserido ou alterado. Possui dois atributos próprios:

*cite* = define a fonte do motivo pelo qual o texto foi excluído.

*datetime* = define quando o texto foi excluído.

**<dfn>:** tag usada para definir a instância de definição de um termo. A instância de definição normalmente é o primeiro uso de um termo.

**<em>:** tag usada para definir uma ênfase em parte de um texto.

**<hr>:** (horizontal row) tag que NÃO possui fechamento, usada para definir uma quebra temática na página. Normalmente, é usada para separar o conteúdo.

**<i>:** tag usada para definir parte de um texto em uma voz ou humor diferente (para leitores de tela). O conteúdo é exibido em itálico e pode indicar um termo técnico, uma frase em outro idioma ou um pensamento. Antes de usar essa tag, deve-se considerar o uso das tags <em>, <strong>, <mark>, <cite> e <dfn>.

**<kbd>:** tag usada para definir uma tecla do teclado em uma frase.

**<mark>:** tag usada para definir marcar um texto (destacar partes do texto).

**<ruby>:** tag usada para definir uma anotação em ruby. Anotação ruby é um pequeno texto extra, anexado ao texto principal, para indicar a pronúncia ou o significado dos caracteres correspondentes. Muito usada em publicações japonesas.

**<rp>:** tag usada para definir parênteses em navegadores que não suportam tags de anotação <ruby>.

**<rt>:** tag usada para definir um texto ruby em uma anotação ruby. A tag fornece informações de pronúncia, tradução ou transliteração do Leste Asiático. Sempre deve ser usada dentro de uma tag <ruby>.

**<s>:** tag usada para definir um texto que não é mais correto, preciso ou relevante. Define um risco em cima do texto, como a tag <del>, porém o texto em questão não foi removido ou excluído para o uso dessa outra tag.

**<small>:** tag usada para definir um texto pequeno.

**<strong>:** tag usada para definir um texto como importante.

**<sub>:** tag usada para definir um texto subscrito. Normalmente usada para fórmulas químicas.

**<sup>:** tag usada para definir um texto sobrescrito. Normalmente usada para notas de rodapé ou expoentes matemáticos.

**28/11 – TAGS (PARTE 8)**

**<time>:** tag usada para definir uma data e/ou hora legível para humanos. Possui um atributo próprio:

*datetime* = define a data e/ou hora visível legível também para leitores de máquina.

**<u>:** tag usada para definir uma parte de um texto sublinhado. Usado normalmente para mostrar que uma palavra está com erro de ortografia. Cuidado ao usar essa tag, para não confundir o usuário com um hiperlink. Possui peso semântico.

**<pre>:** tag usada para definir um texto pré-formatado. Esta tag preserva os espaços e quebras de linha.

**<samp>:** define um texto como saída de amostra de um programa de computador. Muito usada para indicar mensagens de erro do computador (respostas da máquina).

**<var>:** vem da palavra “variable”, que significa variável. Tag usada para definir um texto como variável.

**<code>:** tag usada para definir parte de um texto como um código de computador.

**<dl>:** (description list) tag usada para definir uma lista com descrição. É usada em conjunto com as tags <dt> e <dd>.

**<dt>:** (description term) tag usada para definir um termo em uma lista com descrição. É usada em conjunto com as tags <dl> e <dd>.

**<dd>:** (description details) tag usada para definir a descrição de um termo. É usada em conjunto com a tag <dt> e deve estar dentro de uma tag <dl>. A tag pode ter em seu conteúdo parágrafos, quebras de linha, imagens, links, listas, etc.

**<form>:** tag usada para definir um formulário. Os formulários agrupam campos de entrada de dados dos usuários. Possui oito atributos próprios:

*accept-charset* = define o encode usado ao enviar o formulário.

*action* = define para qual URL o formulário será enviado.

*autocomplete* = define se os campos terão a função auto completar habilitados.

*enctype* = define qual codificação de formulário será usada.

*method* = define qual será o método de envio do formulário. (get ou post (mais comum))

*name* = define qual o nome do formulário.

*novalidate* = define se deve ser validado ao enviar.

*target* = define qual o alvo ao enviar o formulário.

**<input>:** tag que NÃO tem fechamento, usada para definir um campo de entrada de dados pelo usuário. Esse campo de entrada serve para o usuário inserir dados ou interagir com a página. A tag pode variar de várias maneiras dependendo do atributo *type*. Possui diversos atributos próprios:

*accept* = define o tipo de arquivo aceito para ser selecionado. (se aplica apenas quando o atributo *type* tem o valor *file*)

*alt* = define um texto alternativo. (se aplica apenas quando o atributo *type* tem o valor *image)*

*autocomplete* = define se o campo pode ter habilitado o auto completar do navegador ou nào.

*autofocus* = define o foco neste campo quando a página é carregada.

*checked* = define se o campo está marcado/selecionado. (se aplica apenas quando o atributo *type* tem o valor *checkbox* ou *radio*)

*dirname* = define o nome do campo que será enviado informando a direção do texto escrito.

*disabled* = define o campo como desabilitado.

*form* = define qual o formulário que esse campo pertence.

*formaction* = define qual a URL de envio desse campo. (se aplica apenas quando o atributo *type* tem o valor *submit* ou *image*)

*formenctype* = define qual a enctype do formulário ao enviar esse campo. (se aplica apenas quando o atributo *type* tem o valor *submit* ou *image*)

*formmethod* = define qual será o método do formulário ao clicar neste botão. (se aplica apenas quando o atributo *type* tem o valor *submit* ou *image*)

*formnovalidate* = desativa a validação do formulário ao clicar neste botão.

*formtarget* = define qual será o target do formulário ao clicar neste botão. (se aplica apenas quando o atributo *type* tem o valor *submit* ou *image*)

*height* = define a altura do campo. (se aplica apenas quando o atributo *type* tem o valor *image*)

*list* = define qual tag *datalist* está vinculada com este campo.

*max* = define um valor máximo. (se aplica apenas quando o atributo *type* tem o valor *number* ou *date*)

*maxlength* = define um comprimento máximo de caracteres. (se aplica apenas quando o atributo *type* tem o valor *number*)

*min* = define um valor mínimo. (se aplica apenas quando o atributo *type* tem o valor *number* ou *date*)

*multiple* = define se o usuário pode entrar com mais de um valor neste campo. (se aplica apenas quando o atributo *type* tem o valor *file*)

*name* = define o nome do campo.

*pattern* = define uma expressão regular para a validação do campo.

*placeholder* = define um texto de dica que descreve o valor esperado.

*readonly* = define o campo como somente leitura.

*required* = define o campo como obrigatório para o envio do formulário.

*size* = define a largura, em caracteres, do campo.

*src* = define a fonte do arquivo. (se aplica apenas quando o atributo *type* tem o valor *image*)

*step* = define o intervalo de números permitidos. (se aplica apenas quando o atributo *type* tem o valor *number*)

*type* = define o tipo de campo. (button, checkbox, color, date, datetime-local, email, file, hidden, image, month, number, password, radio, range, reset, search, submit, tel, text, time, url, week)

*value* = define o valor inicial do campo.

*width* = define a largura do campo. (se aplica apenas quando o atributo *type* tem o valor *image*)

**<button>:** tag usada para definir um botão clicável. Dentro desta tag, pode ser colocado conteúdo como texto ou imagem. É importante sempre especificar o atributo type. Possui onze atributos próprios:

*autofocus* = define o foco neste elemento automaticamente ao carregar a página.

*disabled* = define o botão como desabilitado.

*form* = define a que formulário este botão pertence.

*formaction* = define qual será o action do formulário ao clicar neste botão.

*formenctype* = define qual será o enctype do formulário ao clicar neste botão.

*formmethod* = define qual será o método do formulário ao clicar neste botão.

*formnovalidate* = desativa a validação do formulário ao clicar neste botão.

*formtarget* = define qual será o target desse formulário ao clicar nesse botão.

*name* = define o nome do botão.

*type* = define o tipo do botão. (button, reset, submit)

*value* = define o valor inicial do botão.

**29/11 – TAGS (PARTE 9)**

**<fieldset>:** tag usada para definir um grupo de campos de formulário. Esta tag desenha uma caixa ao redor dos campos agrupados. Pode ter uma tag <legend> em seu conteúdo, além dos campos. Possui três atributos próprios:

*disabled* = define o conteúdo agrupado como desabilitado.

*form* = define a qual formulário este grupo de campos pertence.

*name* = define o nome do grupo de campos

**<legend>:** tag usada para definir uma legenda para a tag <fieldset>.

**<select>:** tag usada para definir um campo de lista suspensa. Cada opção do campo é definida por uma tag <option>. Possui sete atributos próprios:

*autofocus* = define o foco neste campo quando a página é carregada.

*disabled* = define o campo como desabilitado.

*form* = define qual o formulário que esse campo pertence.

*multiple* = define se o usuário pode selecionar mais de uma opção.

*name* = define o nome do campo.

*required* = define o campo como obrigatório para o envio do formulário.

*size* = define quantas opções devem ficar visíveis na lista.

**<option>:** tag usada para definir uma opção em uma lista suspensa. Possui quatro atributos próprios:

*disabled* = define um grupo como desabilitado.

*label* = define o rótulo do grupo.

*selected* = define se a opção deve ser selecionada por padrão.

*value* = define o valor que será enviado caso a opção seja selecionada.

**<optgroup>:** tag usada para agrupar opções relacionadas em uma lista suspensa. Possui dois atributos próprios:

*disabled* = define um grupo como desabilitado.

*label* = define o rótulo do grupo.

**<datalist>:** tag usada para definir uma lista de opções para um <input>. O ID desta tag é usado no atributo *list* da tag <input> para vinculação.

**<textarea>:** tag usada para definir um campo de entrada de texto com várias linhas. Possui vários atributos próprios:

*autofocus* = define o foco neste campo quando a página é carregada.

*cols* = define largura visível.

*dirname* = define o nome do campo que será enviado informando a direção do texto escrito.

*disabled* = define o campo como desabilitado.

*form* = define qual o formulário que esse campo pertence.

*maxlength* = define um comprimento máximo de caracteres.

*name* = define o nome do campo.

*placeholder* = define um texto de dica que descreve o valor esperado.

*readonly* = define o campo como somente leitura.

*required* = define o campo como obrigatório para o envio do formulário.

*rows* = define a quantidade de linhas visíveis do campo.

*wrap* = define como o texto deve ser quebrado quando enviado.

**<dialog>:** tag usada para definir uma caixa de diálogo. Esta tag facilita a criação de um modal na página. Possui um atributo próprio:

*open* =define se a caixa de diálogo deve estar visível.

**<output>:** tag usada para mostrar o resultado de um cálculo. Possui três atributos próprios:

*for* = define a relação entre o resultado e os elementos usados para o cálculo.

*form* = define qual o formulário que esse campo pertence.

*name* = define o nome do elemento.

**30/11 – TAGS (PARTE 10)**

**<meter>**: tag usada para definir uma medida dentro de um intervalo. Possui sete atributos próprios:

*form* = define a qual formulário esse medidor pertence.

*high* = define a partir de qual valor deve ser considerado um valor alto.

*low* = define até qual valor deve ser considerado um valor baixo.

*max* = define o valor máximo.

*min* = define o valor mínimo.

*optimum* = define o valor ideal para o medidor.

*value* = define o valor atual.

**<progress>:** tag usada para definir o nível de progresso de uma tarefa. Possui dois atributos próprios:

*max* = define o valor máximo do progresso.

*value* = define o valor completado do progresso.

**<canvas>:** tag usada para definir um quadro de desenhos gráficos 2D ou 3D. Para desenhar, é necessário utilizar JavaScript. Possui dois atributos próprios:

*height* = define a altura do seu quadro.

*width* = define a largura do seu quadro.

**<svg>:** tag usada para definir um contêiner para gráficos vetoriais escaláveis. Possui dois atributos próprios:

*height* = define a altura do seu contêiner.

*width* = define a largura do seu contêiner.

**<hgroup>:** tag usada para agrupar as tags <h1>, <h2>, <h3>, <h4>, <h5> e <h6>.

**<marquee>:** tag usada para definir uma área de rolagem de texto. Possui sete atributos próprios:

*behavior* = define como o texto será rolado. (scroll, slide, alternate)

*direction* = define qual será a direção. (left, right, up, down)

*height* = define a altura da área.

*loop* = define o número de vezes que a rolagem será rolada.

*scrollamount* = define a quantidade de rolagem em cada intervalo.

*scrolldelay* = define o intervalo entre cada movimento de rolagem.

*width* = define a largura da área.

**01/12 – EVENTOS (PARTE 1)**

**Eventos** são gatilhos disparados devido alguma interação. A maioria dos gatilhos são disparados por ações do usuário. Normalmente, executamos algum código JavaScript após um evento acontecer. Temos eventos de: janela, formulário, teclado, mouse, arrasto, transferência, mídia e outros.

Nos atributos HTML, todo evento deve ter o prefixo “on”.

A seguir, os eventos de **janela:**

*afterprint* = disparado depois da página ser impressa.

*beforeprint* = disparado antes da página ser impressa.

*beforeunload* = disparado antes do navegador carregar outra página.

*error* = disparado quando ocorrer um erro.

*hashchange* = disparado quando o hash da URL (#) da página é alterado.

*load* = disparado quando toda a página for carregada.

*message* = disparado quando ocorrer uma mensagem.

*offline* = disparado quando o navegador é desconectado da internet.

*online* = disparado quando o navegador é conectado com a internet.

*pagehide* = disparado quando a página é ocultada.

*pageshow* = disparado quando a página é mostrada.

*popstate* = disparado quando a ocorrência no histórico do navegador é alterada.

*resize* = disparado quando a janela é redimensionada.

*storage* = disparado quando o armazenamento do navegador é alterado.

*unload* = disparado quando o usuário atualiza a página ou fecha a janela.

A seguir, os eventos de **formulário:**

*blur* = disparado quando o foco do elemento é removido.

*change* = disparado quando termina de alterar o valor do elemento.

*contextmenu* = disparado quando o usuário abre o menu de contexto.

*focus* = disparado quando o elemento é focado.

*input* = disparado quando o elemento recebe uma entrada de dados.

*invalid* = disparado quando o valor do elemento é inválido.

*reset* = disparado quando o valor do elemento é redefinido ao estado inicial.

*search* = disparado quando o elemento input do tipo search é enviado.

*select* = disparado quando o valor do elemento é selecionado.

*submit* = disparado quando o formulário é enviado.

A seguir, os eventos de **teclado**:

*keydown* = disparado quando a tecla vai para baixo.

*keypress* = disparado quando a tecla é pressionada.

*keyup* = disparado quando a tecla é solta.

A seguir, os eventos de **mouse**:

*click* = disparado quando o elemento recebe um clique.

*dbleclick* = disparado quando o elemento recebe um clique duplo.

*mousedown* = disparado quando o botão de clique vai para baixo.

*mousemove* = disparado quando o cursor do mouse se move sobre o elemento.

*mouseout* = disparado quando o cursor do mouse não está em cima do elemento.

*mouseover* = disparado quando o cursor do mouse está em cima do elemento.

*mouseup* = disparado quando o botão do clique é solto.

*mousewheel* = disparado quando a roda do mouse (scroll) é rodada.

*mouseenter* = disparado quando o cursor do mouse entra em cima do elemento.

*mouseleave* = disparado quando o cursor do mouse sai de cima do elemento.

A seguir, os eventos de **arrasto**:

*drag* = disparado quando o elemento é arrastado.

*dragend* = disparado quando o elemento para de ser arrastado.

*dragenter* = disparado quando o elemento arrastado entra em um elemento que permite soltar.

*dragleave* = disparado quando o elemento arrastado sai de cima de um elemento que permite soltar.

*dragover* = disparado quando o elemento arrastado passa em um elemento que permite soltar.

*dragstart* = disparado quando o elemento começa a ser arrastado.

*drop* = disparado quando o elemento é solto.

*scroll* = disparado quando o scroll é rolado.

A seguir, os eventos de **transferência:**

*copy* = disparado quando o conteúdo do elemento é copiado.

*cut* = disparado quando o conteúdo do elemento é recortado.

*paste* = disparado quando o conteúdo do elemento é colado.

A seguir, os eventos de **mídia**:

*abort* = disparado quando o carregamento da mídia é abortado.

*canplay* = disparado quando a mídia carregou o suficiente para começar.

*canplaythrough* = disparado quando a mídia carregou e pode ser reproduzida até o final.

*cuechange* = disparado quando o texto de uma trilha é alterado.

*durationchange* = disparado quando a duração total da mídia é alterada.

*emptied* = disparado quando acontece algum problema e a mídia fica indisponível.

*ended* = disparado quando a mídia foi consumida até o fim.

*error* = disparado quando um erro acontece.

*loadeddata* = disparado quando os dados da mídia são carregados.

*loadedmetadata* = disparado quando os metadados da mídia são carregados. (Os metadados na mídia são: informações de duração, título, nome do arquivo, etc)

*loadstart* = disparado quando os dados da mídia começam a ser carregados.

*pause* = disparado quando a mídia é pausada.

*play* = disparado quando a mídia é começada.

*playing* = disparado depois que a mídia realmente começou.

*progress* = disparado enquanto o navegador obtém os dados da mídia (download).

*ratechange* = disparado quando a taxa de reprodução é alterada.

*seeked* = disparado quando a busca de dados da mídia terminou.

*seeking* = disparado quando a busca de dados da mídia está acontecendo.

*stalled* = disparado quando o navegador não pode buscar os dados de mídia por qualquer motivo.

*suspend* = disparado quando a busca de dados de mídia é parada antes de ser completamente carregada.

*timeupdate* = disparado quando a posição da reprodução é alterada.

*volumechange* = disparado quando o volume do áudio da mídia é alterado.

*waiting* = disparado quando a mídia é pausada para armazenar mais dados em buffer.

Outro evento sem categoria:

*toggle* = disparado quando o usuário abre ou fecha o elemento details.