GUIA COMPLETO DESCUBRA O PODER DO HTML

Por Alisson Suassuna

Guia Completo Descubra o Poder do HTML

Por Alisson Suassuna

ISENÇÃO DE RESPONSABILIDADE

Todas as informações contidas neste guia são provenientes de minhas experiências pessoais ao longo de vários anos. Embora eu tenha me esforçado ao máximo para garantir a precisão e a mais alta qualidade dessas informações e acredite que todas as técnicas e métodos aqui ensinados sejam altamente efetivos, eu não me responsabilizo por erros ou omissões. Sua situação e/ou condição particular pode não se adequar perfeitamente aos métodos e técnicas ensinados neste guia. Assim, você deverá utilizar e ajustar as informações deste guia de acordo com sua situação e necessidades.

Todos os nomes de marcas, produtos e serviços mencionados neste guia são propriedades de seus respectivos donos e são usados somente como referência.

DIREITOS AUTORAIS

Um Pouco Mais Sobre Meu Trabalho

Vamos nos conhecer melhor? Então vamos bater um papo... segue abaixo meus principais contatos, espero você !

- **Meu Blog Pessoal sobre Desenvolvimento FullStack**
  - **TipsCode.com.br**
- **Meu Canal De Vídeos no Youtube:**
  - **Youtube.com/TipsCode**
- **Minha Página No Facebook:**
  - **Facebook.com/tipsCodeOficial**
- **Meu Perfil No Instagram**
  - **Instagram/TipsCodeOficial**

Conheça Aqui

Sumário

Olá Tudo bem?

... Meu nome é **Alisson Suassuna**, sou desenvolvedor fullstack e nessa **rápida e inteligente leitura** vou te mostrar um **caminho claro e real de como você pode fazer para ser tornar um programador ou programadora de sucesso**.

Nesse ebook vou te dar uma visão geral, algumas dicas e um passo a passo simples para que voce possa começar, e na verdade saber que um dos pilares de um programador **fullstack** de sucesso é ter total dominio sobre a linguagem HTML.

Assim como uma **visão clara a respeito da linguagem** e como ela é **poderosa** quando usada **corretamente**.

Porque Se torna um Desenvolvedo(a)r

Hoje eu trabalho dentro da minha casa... essa profissão ela pode ser realizada 100% dentro do conforto do seu lá, basta você ter um computador ou notebook com acesso a internet.

Então, pare por um minuto e imagine: levantar pela manhã, fazer o que tiver que fazer, dar uns 8 passos e chegar no seu escritório...

Sem ninguém ao seu lado te perturbando, te enchendo... oi ainda sem transito, ônibus, chuva...

Para mim isso é o máximo! Esse é um dos poderes de ser tornar um bom desenvolvedor. (Claro que existem aqueles que preferem trabalhar em uma empresa formal e não há nada de errado com isso)

Além disso, como se não bastasse é totalmente indiferente, eu estar aqui em casa ou em qualquer lugar do mundo, tendo uma conexão com internet, **pronto**, posso trabalhar.

... é ótimo!

Área dinâmica e Interativa

Significa que a programação é uma área onde você tem oportunidade de ter contato e interagir com diversas áreas.

Ou seja você não precisa só ficar sentado na frente do computador fazendo a mesma coisa sempre.

Pois o desenvolvimento é algo que pode ser aplicado a praticamente todas as áreas e você como desenvolvedor precisará entender da ao menos um pouco da área que irá desenvolver para realizar seu trabalho.

Necessidade de Mercado

Atualmente não é segredo que a tecnologia e a programação se encontram em praticamente todos os lugares.

Podendo ser no restaurante Japonês que onde você pede pelo tablet, no terminal que te mostra o tempo até o próximo ônibus chegar, nos aparelhos eletrônicos, etc.

Meu ponto é que por essa observação rápida podemos avaliar que a demanda de desenvolvedores é crescente e além do mercado necessitar desses profissionais, caso você entre como desenvolvedor em uma empresa o salário pode ser bem interessante (dependendo do cargo).

Crie seu Próprio Negócio

Exatamente, você sabendo um pouco de desenvolvimento é mais do que capaz de criar sua própria empresa mesmo sozinho inicialmente.

Você já ouviu falar de **startups**, onde a partir da sua ideia é possível adquirir patrocinadores, parceiros foi dessa forma que nasceu empresas como Microsoft, Apple, Facebook.

Você não precisa de muito para começar sua empresa nessa área, somente com seu computador e uma ideia suas possibilidades são ilimitadas.

Liberdade

Esse é absolutamente meu favorito, a liberdade, você ter opções capacidade de escolha, de administrar e trabalhar da maneira que você quiser.

Em muitos trabalhos relacionados a programação, você pode realizar a distância, trabalhar de qualquer local, muitas empresas já permitem o trabalho em casa ou possuem um “FlexTime” (Horário de Trabalho Flexível).

Também existe a possibilidade de trabalhar como Freelancer e você fazer sua rotina, você decidir seu horário de acordar, sua hora trabalho, de almoço, de ir embora, você é seu chefe.

Não precisa de Faculdade

Basta um computador e conexão com a internet para você aprender tudo o que aprenderia na faculdade por conta própria sem perder tempo com provas e trabalhos, pois há muitos recursos online para quem quer entrar nessa área.

As empresas se interessam no seu conhecimento e experiência, e não se você possui um diploma.

O Salário é Excelente

Um estagiário de programação não ganha mais que o de outras profissões, mas assim que ele é efetivado o salário geralmente duplica ou triplica. Esse processo de ter grandes aumentos de salário é comum no começo da carreira,

Onde você pode começar com R$1000,00, no ano seguinte já estar com R$3000,00 e no próximo ano R$5000,00.

Além disso, a responsabilidade de um salário alto não é tão grande como a de outras áreas, onde é necessário ser um gerente, um engenheiro ou um médico para ganhar bem. Vidas não são perdidas quando você erra.

Não fica Desempregado

Mesmo em tempos de crise, a área de tecnologia foi uma das poucas que continuou a crescer.

O número de vagas é grande e variado, pois um projeto envolve diversas pessoas e áreas de conhecimento, e empresas de grande porte podem ter mais de 100 projetos acontecendo ao mesmo tempo.

É praticamente impossível ficar desempregado por muito tempo estando próximo aos grandes centros de tecnologia (São Paulo, Campinas, Rio de Janeiro e demais capitais).

Pode Trabalhar em Casa

Já que as únicas coisas que você precisa para trabalhar são um computador e conexão com a internet, não é necessário ir para o escritório para fazer a mesma coisa que você pode fazer da sua casa.

O home-office ainda não é tão comum, mas tem se popularizado, pois as empresas perceberam que podem diminuir custos, e os programadores não querem perder uma hora ou mais no trânsito sem um bom motivo.

Trabalhar de casa também permite trabalhar para qualquer empresa do mundo e não apenas da sua região.

Faz Novas Amizades

Programador não é antissocial. A maioria tende a gostar de uma ou mais coisas do universo nerd como games, quadrinhos, animes, etc., e sendo assim, trabalhar com essas pessoas é também uma possibilidade de fazer novos amigos, e não apenas colegas de trabalho, que compartilham dos mesmos gostos que você.

Sobe na Carreira Rapidamente

Por ser uma área técnica, os profissionais que sabem usar seu conhecimento para resolver problemas rapidamente e de maneira efetiva conseguem subir de cargo (e salário) rapidamente.

Ao contrário de profissões onde o crescimento é lento e depende mais de tempo e politicagem, a área de programação valoriza um bom profissional, pois se não fizer isso é muito fácil trocar de emprego para uma que valorize.

Calma, Calma... Você Consegue!!

“Ah mas isso é muito difícil..., isso não é pra mim..., eu não tenho experiência..., eu não sei por onde começar... eu não tenho como investir...”

Bom, eu escuto isso quase todos os dias, **e todos os dias provo totalmente o contrário**:

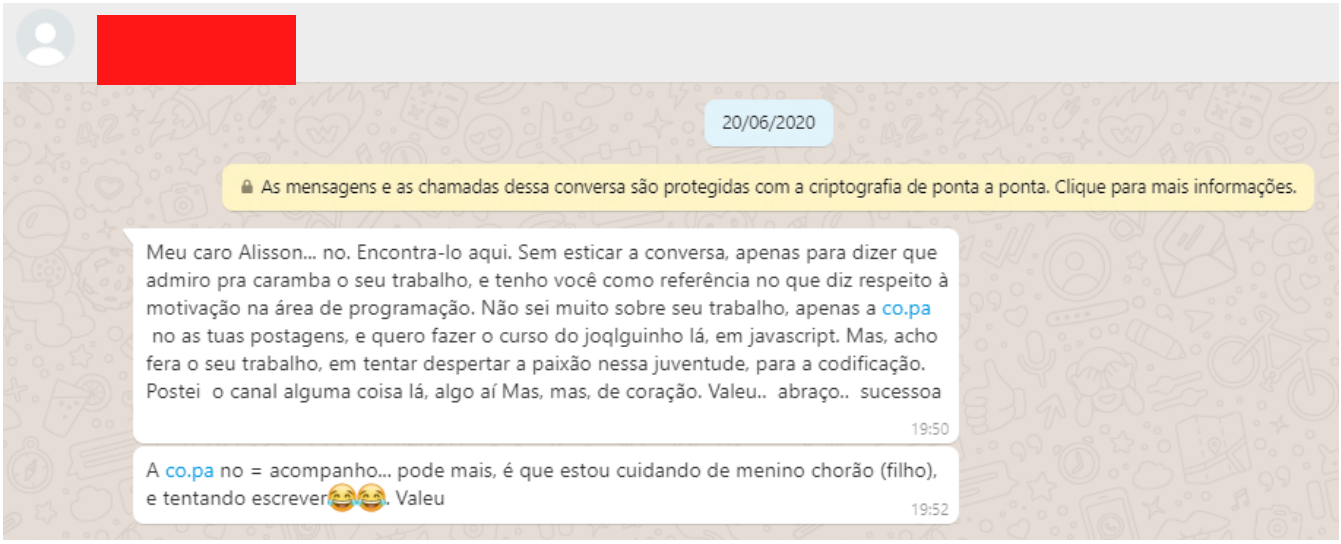
Você não precisa de experiência para se tornar um programador fullstack: Pode parecer complicado, porém é muito simples.

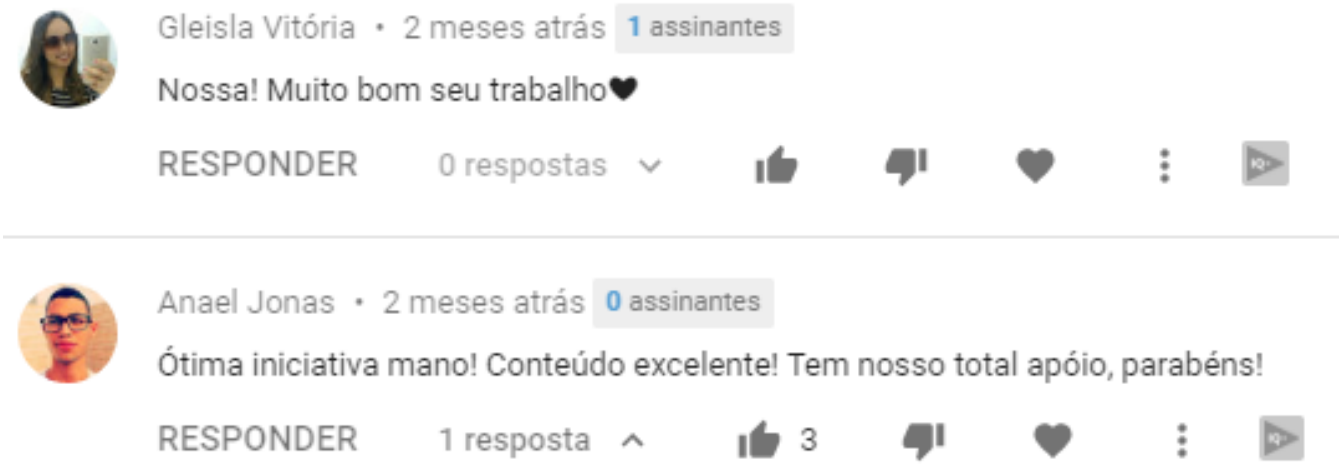
Tem muitos por aí que tentam complicar, ou tentam fazer a coisa complicada (e falam isso para as pessoas) apenas para parecerem mais inteligentes... mas na verdade é muito simples.

O investimento para começar é quase nulo...
... veja no meu caso, eu comecei só com minha força de vontade... não tinha nenhum dinheiro.

Por esse motivo criei o TipsCode onde busco lhe ensinar de maneira muito simples.

Da uma olhada no que alguns escreveram para mim:





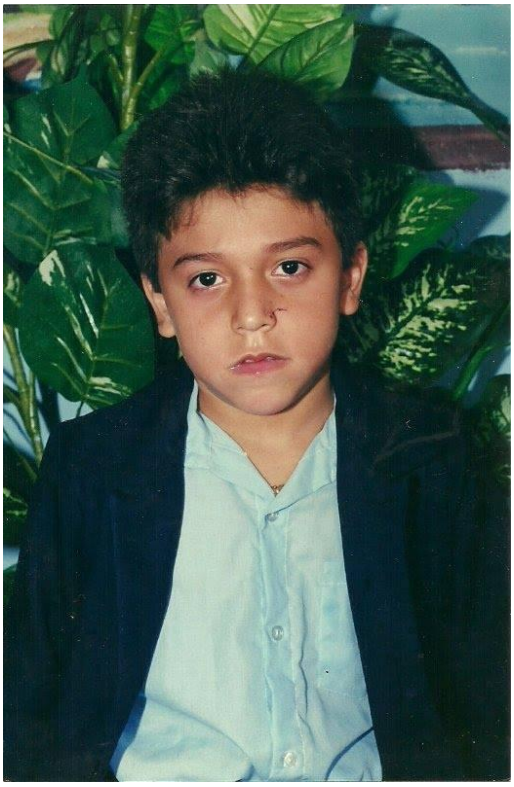
Existe Sim Um Caminho

Olha, eu sei bem como é no início, **ficar totalmente perdido sem saber o que fazer**...

Quando eu comecei foi exatamente assim, **passei mais de 4 anos tentando, tentando e tentando e nada de resultados**.

Ao contrário do que muitos pensam, **não foi nada fácil**, se eu fosse te contar tudo o que eu passei para chegar até aqui, pode pegar um lenço e um balde que ia começar a choradeira...

Eu vim de uma família muito humilde… nunca me faltou nada, mas era complicado...



E eu não quero entrar em detalhes desse ponto da história agora, mas eu perdi meu pais na minha pré-adolescência, e fale o que quiser, não importa o quão bom são as pessoas ao seu redor, pai é pai e ponto…

Bom, esse acontecimento mexeu muito comigo, mas chegou um ponto que comecei meus estudos muito fortes, tive alguns empregos, várias tentativas de pequenos negócios, nunca parei de estudar, mas sempre me virei, com a graça de Deus...

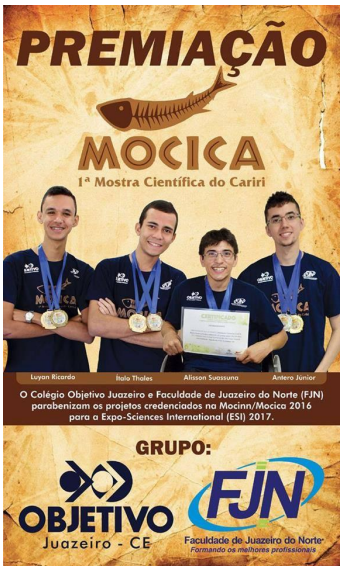
Eu sou natural do Rio Grande Do Norte, e lá no inverno é frio, mas frio mesmo...

A hora Da Virada..

Conheci a Ana Paula que hoje é minha esposa, vim mora no Ceará e aqui foi onde tiver a oportunidade de conhece bem a área da programação, Deus me deu a oportunidade de cursar o curso superior de Bacharelado em Sistemas de Informação, onde tive várias oportunidades de empregos, estágios.

Fui Monitor da disciplina de Tecnologias web por 2 anos, conhecia muito programadores bons, empresarios, fiz algumas viagens para congressos como por exemplo a Campus Patty.

Eu e mais dois colegas desenvolvemos um projeto onde criamos um Sistema Computacional de Baixo Custo para comunidades carentes, onde esse projeto foi muito premiado em algumas feiras de ciências e com esse projeto participei da (ESI) Expo-Sciences International 2017



Capítulo 02: Introdução ao HTML

Versão	Especificação	Data de laçamento:
1.0	N/A	01/01/1994
2.0	RFC 1866	24/11/1995
3.2	W3C: HTML	14/01/1997
4.0	W3C: HTML	24/04/1998
5.0	WHATWG: HTML	28/01/2014
5.1	W3C: HTML	01/11/2016

Hello World (Olá Mundo)

Introdução

HTML (**H**yper**t**ext **M**arkup **L**anguage) usa um sistema de marcação composto por elementos que representam um conteúdo específico. A marcação significa que, com HTML, você declara o que é apresentado a um navegador. As representações visuais são definidas pelo CSS (Cascading Style Sheets) e realizadas pelos navegadores. Às vezes, o HTML é chamado de **linguagem de programação**, mas não tem lógica, assim como uma linguagem de marcação. As tags HTML fornecem significado semântico e legibilidade da máquina para o conteúdo da página.

Um elemento geralmente consiste em uma tag de abertura (<**nome_elemento**>), e uma tag de fechamento (</**nome_elemento**>), que contém o nome do elemento entre colchetes angulares e o conteúdo entre as duas tags por exemplo: <**nome_elemento**> ... Conteúdo ... </**nome_elemen**to>

Existem alguns elementos HTML que não possuem uma marca de fechamento ou qualquer conteúdo. Estes são chamados de elementos **nulos**. Os elementos nulos incluem <**img**>, <**meta**>, <**link**> e <**input**>. Os nomes dos elementos podem ser considerados palavras-chave descritivas para o conteúdo que eles contêm, como vídeo, áudio, tabela, rodapé.

Uma página HTML pode consistir em centenas de elementos potencialmente lidos por um navegador da Web, interpretados e renderizados em conteúdo legível ou audível por leitores ea tela.Para este documento, é importante observar a diferença entre elementos e tags:

Elementos: vídeo, áudio, table, footer

Tags: <video>, <audio>, <table>, <footer>, </html>, </body>

Inspeção de elementos
Vamos explicar uma tag... A tag <**p**> representa um parágrafo comum.

Os elementos geralmente têm uma marca de abertura e uma marca de fechamento. A tag de abertura contém o nome do elemento em

colchetes (<**p**>). A tag de fechamento é idêntica à tag de abertura com a adição de uma barra (/) entre o colchete de abertura e o nome do elemento (</**p**>). O conteúdo pode ir entre essas duas tags: <**p**> Este é um parágrafo simples. </**p**>.

Criando uma página simples

O exemplo de HTML a seguir cria uma página da Web simples "Hello World".
Arquivos HTML podem ser criados usando qualquer **editor de texto**. Os arquivos devem ser salvos com uma extensão **.html** ou **.htm** para serem reconhecidos como arquivos HTML. Uma vez criado, esse arquivo pode ser aberto em qualquer navegador da web.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <title>Olá, Alisson</title>
</head>
<body>

    <h1>Olá Mundo</h1>
    <p>Estou aprendendo HTML com Alisson Suassuna</p>

</body>
</html>
```

Vamos Passo A Passo.

Estas são as tags usadas no exemplo acima:

Tag

<**! DOCTYPE**> Define a versão do HTML usada no documento. Nesse caso, é **HTML5**.

<**html**> Abre a página. Nenhuma marcação deve ocorrer após a tag de fechamento (</**html**>). O atributo **lang** declara o idioma principal da página.

<**head**> Abre a seção principal, que não aparece na janela principal do navegador, mas contém principalmente informações sobre o documento **HTML**, chamado **metadados**. Também pode conter importações de folhas de estilo (CSS) e scripts externos. A **tag** de fechamento é </**head**>.

<**meta**> Fornece ao navegador alguns **metadados** sobre o documento. O atributo **charset** declara a codificação de caracteres. Documentos HTML modernos sempre devem usar **UTF-8**, mesmo que isso não seja um requisito. Em HTML, a tag <**meta**> não requer uma tag de fechamento.

<**title**> O título da página. O texto escrito entre esta abertura e a tag de fechamento (</**title**>) será exibido na guia da página ou na barra de título do navegador.

<**body**> Abre a parte do documento exibida aos usuários, ou seja, todo o conteúdo visível ou audível de uma página. Nenhum conteúdo deve ser adicionado após a tag de fechamento </**body**>.

<**h1**> Um cabeçalho de nível 1 para a página. Veja os títulos para mais informações.

<**p**> Representa um parágrafo comum do texto.

Capítulo 03: Doctypes

Doctypes - abreviação de 'tipo de documento' - ajuda os navegadores a entender a versão do HTML em que o documento está gravado para melhor interpretação. As declarações de tipo de documento não são tags HTML e pertencem à parte superior de um documento. Este tópico explica a estrutura e a declaração de vários **doctypes** em HTML.

Adicionando o Doctype

A declaração <**! DOCTYPE**> deve sempre ser incluída na parte superior do documento HTML, antes da tag <html>.

Versão ≥ 5

```
<!DOCTYPE html>
```

HTML 5 Doctype

O **HTML5** não é baseado em **SGML** (*Standard Generalized Markup Language*) e, portanto, não requer uma referência a uma **DTD** (*Definição de Tipo de Documento*).

Declaração de tipo de documento HTML 5:

```
<!DOCTYPE html>
```

AVISOS

De acordo com a especificação HTML 5 DOCTYPE do W3.org:

> Um **DOCTYPE** deve consistir nos seguintes componentes e seguir essa ordem:
>
> > Uma sequência que não corresponde a maiúsculas de minúsculas ASCII para a sequência "<**! DOCTYPE**".

portanto, os seguintes DOCTYPEs também são válidos:

```
<!doctype html>
<!dOCtyPe html>
<!DocTYpe html>
```

Capítulo 04: Cabeçalhos

O HTML fornece não apenas tags de parágrafo simples, mas seis tags de cabeçalho separadas para indicar títulos de vários tamanhos e espessuras. Enumerados como cabeçalho 1 a 6, o cabeçalho 1 tem o texto maior e mais grosso, enquanto o cabeçalho 6 é o menor e o mais fino, até o nível do parágrafo. Este tópico detalha o uso adequado dessas tags.

Usando Títulos

Os títulos podem ser usados para descrever o tópico precedente e são definidos com as tags <**h1**> a <**h6**>. Os títulos suportam todos os atributos globais.

- <**h1**> define o cabeçalho mais importante.
- <**h6**> define o cabeçalho menos importante.

Definindo um cabeçalho:

```
<h1>Título 1</h1>
<h2>Título 2</h2>
<h3>Título 3</h3>
<h4>Título 4</h4>
<h5>Título 5</h5>
<h6>Título 6</h6>
```

A estrutura correta é importante

Os mecanismos de pesquisa e outros agentes do usuário geralmente indexam o conteúdo da página com base em elementos de cabeçalho, por exemplo, para criar um índice, portanto, é importante usar a estrutura correta para os cabeçalhos. Em geral, um artigo deve ter um elemento **h1** para o título principal, seguido pelas legendas **h2** - descendo uma camada, se necessário. Se houver elementos **h1** em um nível superior, eles não deverão ser usados para descrever nenhum conteúdo de nível inferior.

Exemplo de documento (intenção extra para ilustrar hierarquia):

```
<h1>Main title</h1>
<p>Introduction</p>
    <h2>Reasons</h2>
        <h3>Reason 1</h3>
        <p>Paragraph</p>

        <h3>Reason 2</h3>
        <p>Paragraph</p>

    <h2>In conclusion</h2>
    <p>Paragraph</p>
```

Capítulo 05: Parágrafo

Coluna	Descrição
<p>	Define um parágrafo
 	Insere uma única quebra de linha
<pre>	Define o texto pré-formatado

Os parágrafos são o elemento **HTML** mais básico. Este tópico explica e demonstra o uso do elemento de parágrafo em HTML

Parágrafos HTML

O elemento HTML <p> define um parágrafo

```
<p>Esse é um parágrafo..</p>
<p>Outro parágrafo</p>
```

Exibição (Display)

Você não pode ter certeza de como o HTML será exibido.

Telas grandes ou pequenas e janelas redimensionadas criarão resultados diferentes.

Com o HTML, você não pode alterar a saída adicionando espaços ou linhas extras ao seu código HTML.

O navegador removerá espaços e linhas extras quando a página for exibida

```
<p>Este é      outro parágrafo, os      espaços      extras      serão      removidos
pelos          navegadores</p>
```

Capítulo 06: Formatação de texto

Embora a maioria das tags HTML seja usada para criar elementos, o HTML também fornece tags de formatação de texto para aplicar *estilos específicos* relacionadas ao texto em algumas partes do texto. Este tópico inclui exemplos de formatação de texto HTML, como **realce**, **negrito**, **sublinhado**, **subscrito** e texto **destacado**.

Destacando Texto

O elemento <mark> é novo no HTML5 e é usado para marcar ou destacar texto em um documento "devido à sua relevância em outro contexto" . O exemplo mais comum seria nos resultados de uma pesquisa se o usuário inserisse uma consulta de pesquisa e os resultados fossem mostrados destacando a consulta desejada.

```
<p>Aqui está um conteúdo de um artigo que contém o <mark>tudo sobre HTML</mark> que estamos procurando. Destacar o texto tornará mais fácil para o usuário encontrar o que está procurando.</p>
```

Resultado:

Aqui está um conteúdo de um artigo que contém o tudo sobre HTML que estamos procurando. Destacar o texto tornará mais fácil para o usuário encontrar o que está procurando.

Uma formatação padrão comum é o texto em preto sobre fundo amarelo, mas isso pode ser alterado com CSS.

Negrito, Italic e Sublinhado

Texto em negrito com HTML

Para texto em negrito, use as tags ou <b>:

Qual a diferença? Semântica. <**strong**> é usado para indicar que o texto é fundamental ou semanticamente importante para o texto circundante, enquanto <**b**> indica essa importância e simplesmente representa o texto que deve estar em negrito. Se você usasse <**b**>, um programa de conversão de texto em fala não diria a (s) palavra (s) de maneira diferente das outras palavras ao seu redor - você está simplesmente chamando a atenção para elas sem acrescentar nenhuma importância adicional. Ao usar <**strong**>, no entanto, o mesmo programa vai falar essas palavras com um tom de voz diferente para transmitir que o texto é importante de alguma forma.

Texto em itálico

Para colocar o texto em itálico, use as tags ou <i>:

```
<em>Texto em Itálico</em>
```

Ou

Texto em Itálico

Qual a diferença? Semântica. A tag <**em**> é usado para indicar que o texto deve ter uma ênfase extra que deve ser enfatizada, enquanto simplesmente representa o texto que deve ser definido a partir do texto normal ao seu redor.

Por exemplo, se você quiser enfatizar a ação dentro de uma frase, enfatize-a em itálico via : "Você já enviaria a edição?"

Mas se você estivesse identificando um livro ou jornal que normalmente usaria itálico estilisticamente, usaria : "Fui forçado a ler Romeu e Julieta no ensino médio.

Texto Sublinhado

Embora o próprio elemento <**u**> tenha sido descontinuado no **HTMl4**, ele foi reintroduzido com significado semântico alternativo no HTML 5 - para representar uma anotação não-textual e desarticulada. Você pode usar essa renderização para indicar um texto incorreto na página ou para uma marca de nome próprio chinês.

```
<p>Este parágrafo contém alguns <u>textos incoretos</u> etc...</p>
```

Abreviação

Para marcar alguma expressão como uma abreviação, use a tag <abbr>:

```
<p>Estamos estudando <abbr title="Hypertext Markup Language">HTML</abbr>!</p>
```

Se presente, o atributo title é usado para apresentar a descrição completa dessa abreviação.

Inserido, excluído ou detalhar

Para marcar o texto como inserido, use a tag <**ins**>:

```
<ins>Novo texto</ins>
```

Para marcar o texto como excluído, use a tag <del>

```
<del>Texto deletado</del>
```

Para detalhar o texto, use a tag <s>:

```
<s>Texto detalhado aqui</s>
```

Sobrescrito e subscrito

Para definir o texto para cima ou para baixo, você pode usar as tags <sup> e <sub>.

Para criar sobrescrito e Subescrito:

```
<sup>Texto Sobrescrito</sup>
```

Capítulo 07: Âncoras e HiperLinks

Atributos	Descrição
href	Especifica o endereço de destino. Pode ser uma URL absoluta ou relativa ou o nome de uma âncora. A URL absoluta é uma URL completa de um site por exemplo: http://www.tipscode.com.br/. Uma URL relativa aponta para outro diretório e / ou documento dentro do mesmo site, por exemplo / sobre / aponta para o diretório "sobre" dentro do diretório raiz (/). Ao apontar para outro diretório sem especificar explicitamente No documento, os servidores Web normalmente retornam o documento "**index.html**" dentro desse diretório.
hreflang	Especifica o idioma do recurso vinculado pelo atributo href (que deve estar presente com este Use valores de idioma do **BCP 47** para HTML5 e **RFC 1766** para HTML 4.
rel	Especifica o relacionamento entre o documento atual e o documento vinculado. Para HTML5, o Os valores devem ser definidos na especificação.
target	Especifica onde abrir o **link**, por exemplo. em uma nova guia ou janela. Os valores possíveis são **_blank**, **_self**, **_parent**, **_top**. Forçar esse comportamento não é recomendado, pois viola o controle do usuário sobre um site.
title	Especifica informações extras sobre um link. As informações geralmente são mostradas como um texto de dica de ferramenta quando o cursor se move sobre o link. Este atributo não está restrito a links, pode ser usado em quase todos Tags HTML
download	Especifica que o destino será baixado quando um usuário clicar no hiperlink. O valor do O atributo será o nome do arquivo baixado. Não há restrições nos valores permitidos, e o O navegador detectará automaticamente a extensão de arquivo correta e a adicionará ao arquivo (.img, .pdf, etc.). Se o omitido, o nome do arquivo original é usado.

Tags âncora são comumente usadas para vincular páginas da web separadas, mas também podem ser usadas para vincular entre diferentes coloca em um único documento, geralmente no sumário ou até mesmo lança aplicações externos. Este tópico explica a implementação e aplicação de tags âncora HTML em várias funções

Link para outro site

Este é o uso básico do elemento <a> (elemento âncora):

```
<a href="https://www.tipscode,com.br/"> Link para o site TipsCode </a>
```

Ele cria um hiperlink para a URL http://www.tipscode.com.br/ conforme especificado pelo atributo href (referência de hipertexto), com o texto âncora "Link para tipscode.com.br".

Para indicar que esse link leva a um site externo, você pode usar o tipo de link externo:

```
<a href="https://www.tipscode,com.br/" rel="external">  exemplo site TipsCode  </a>
```

Você pode vincular a um site que usa um protocolo diferente de HTTP. Por exemplo, para vincular a um site FTP, você pode fazer,

```
<a href="ftp:https://tipscode,com.br/download" rel="external">Link para um serviço FTP </a>
```

Nesse caso, a diferença é que essa tag âncora está solicitando que o navegador do usuário se conecte a tipscode.com.br/download usando o FTP (File Transfer Protocol) em vez do HTTP (Hypertext Transfer Protocol).

Link para uma Âncora (<a>)

As âncoras podem ser usadas para pular para tags específicas em uma página HTML. A tag <**a**> pode apontar para qualquer elemento que tenha um atributo **ID**. Para saber mais sobre IDs, Falarei de IDs mais frente. As âncoras são usadas principalmente para pular a uma subseção de uma página e são usados em conjunto com as tags de cabeçalho.

Suponha que você tenha criado uma página (page1.html) sobre vários tópicos:

```
<h2>Primeiro Tópico</h2>
<p>Conteúdo do primeiro tópico</p>
<h2>Segundo Tópico</h2>
<p>Conteúdo do segundo tópico</p>
```

Depois de ter várias seções, convém criar um Sumário na parte superior da página com links rápidos (ou favoritos) para seções específicas.

Se você atribuiu um atributo id aos seus tópicos, poderá vincular a eles. Veja o exemplo:

```
<h2 id="topico1">Primeiro Tópico</h2>
<p>Conteúdo do primeiro tópico</p>
<h2 id="topico2">Segundo Tópico</h2>
<p>Conteúdo do segundo tópico</p>
```

Agora você pode usar a âncora no seu índice:

```
<h1> Tabela de Conteúdo </h1>
        <a href="#topico1"> Clique aqui para ir para o Primeiro tópico </a>
        <a href="#topico2"> Clique aqui para ir para o Segundo tópico </a>
```

Link para uma página do mesmo site

Você pode usar um caminho relativo para vincular a páginas no mesmo site.

```
<a href="/tipscode">Clique no Texto</a>
```

O exemplo acima iria para o exemplo de arquivo no diretório raiz (/) do servidor. Se esse link estivesse em http://tipcode.com.br, os dois links a seguir levariam o usuário ao mesmo local

```
<a href="/pagina">Clique no Texto</a>
<a href="http://tipcode.com.br/pagina">Clique no Texto</a>
```

Ambos os itens acima iriam para o arquivo de paginação no diretório raiz de example.com.

Link de Números

Se o valor do atributo href começar com tel:, seu dispositivo discará o número quando você clicar nele. Isso funciona em dispositivos móveis ou computadores / tablets com software - como Skype ou FaceTime - que podem fazer chamadas telefônicas.

```
<a href="tel:(88)9.99888855">Ligar Agora!</a>
```

A maioria dos dispositivos e programas solicitará ao usuário que confirme o número que está prestes a discar.

Link para abrir em uma nova página

```
<a href="tipscode.com.br" target="_blank">Clique no texto</a>
```

O atributo **target** especifica onde abrir o link. Ao defini-lo como **_blank**, você solicita ao navegador que o abra em um nova guia ou janela (por preferência do usuário).

AVISO SOBRE A SEGURANÇA

Usar **target** = "**_ blank**" fornece ao site de abertura acesso parcial ao objeto **window.opener** via **JavaScript**, que permite que a página acesse e altere o **window.opener.location** da sua página e potencialmente redirecionar usuários para sites de **malware** ou **phishing**. Sempre que usar isso para páginas que você não controla, adicione rel = "noopener" ao seu link para impedir que o objeto window.opener seja enviado com a solicitação. Atualmente, o Firefox não suporta noopener, portanto, você precisará usar rel = "noopener noreferrer" para ser mais seguro.

Link que executa via JavaScript

Basta usar o protocolo javascript: para executar o texto como JavaScript em vez de abri-lo como um link normal:

```
<a href="javascript:myFunction()">Execute o Código</a>
```

Você também pode conseguir a mesma coisa usando o atributo onclick:

```
<a href="#" onclick="myFunction(); return false;">Execute o Código</a>
```

O retorno **false**; é necessário para impedir que sua página role para o topo quando o link para # é clicado. Inclua todo o código que você deseja executar antes dele, pois o retorno interromperá a execução de mais códigos. Também digno de nota, você pode incluir um ponto de exclamação ! após a **hashtag** para impedir que a página seja rolando para o topo. Isso funciona porque inválida fará com que o link não role em qualquer lugar da página, porque não foi possível localizar o elemento que faz referência (um elemento com id = "!"). Você também pode usar o **slug** (como #scrollsNowhere) para obter o mesmo efeito. Nesse caso, retorne false; não é necessário:

```
<a href="#!" onclick="myFunction();">Execute o Código</a>
```

AVISO: Você deveria estar usando algo disso?

A resposta é não. A execução de JavaScript embutido com o elemento HTML como este é uma má prática. Considere usar soluções JavaScript puras que procuram o elemento na página e vinculam um função para ele em vez disso. Ouvindo um evento Considere também se esse elemento é realmente um botão em vez de um link. Nesse caso, você deve usar <button>.

Link para abrir clientes de e-mail

Uso básico

Se o valor do atributo href começar com **mailto**: ele tentará abrir um cliente de email ao clicar em:

```
<a href="mailto:suporte@tipscode.com.br">Enviar E-mail</a>
```

Isso colocará o endereço de email suporte@tipscode.com.br como o destinatário do email recém-criado.

Cc e Cco
Você também pode adicionar endereços para destinatários cc ou cco usando a seguinte sintaxe:

```
<a href="mailto:suporte@tipscode.com.br?
cc=suporte2@tipscode.com.br&bcc=alissonsuassuana@gmail.com">Enviar E-mail</a>
```

Assunto e corpo do texto

Você também pode preencher o assunto e o corpo do novo email

```
<a href="mailto:suporte@tipscode.com.br?subject="QualquerCoisa&body=mensagem"">Execute
o Código</a>
```

Clicar em um link com mailto: tentará abrir o cliente de email padrão especificado pelo seu sistema operacional ou peça para você escolher qual cliente deseja usar. Nem todas as opções especificadas após o endereço do destinatário são suportadas em todos os clientes de email.

```
<a href="#!" onclick="myFunction();">Execute o Código</a>
```

AVISO: Você deveria estar usando algo disso?

A resposta é não. A execução de JavaScript embutido com o elemento HTML como este é uma má prática. Considere usar soluções JavaScript puras que procuram o elemento na página e vinculam um função para ele em vez disso. Ouvindo um evento Considere também se esse elemento é realmente um botão em vez de um link. Nesse caso, você deve usar <button>.

Capítulo 08: Elemento Lista (List)

O HTML oferece três maneiras de especificar listas: listas ordenadas, listas não ordenadas e listas de descrição. Listas ordenadas usam Seqüências ordinais para indicar a ordem dos elementos da lista, as listas não ordenadas usam um símbolo definido, como um marcador para listar. elementos em nenhuma ordem designada e listas de descrição usam recuos para listar elementos com seus filhos. Este tópico explica a implementação e combinação dessas listas na marcação HTML.

Lista Ordenada

Uma lista ordenada pode ser criada com a tag <ol> e cada item da lista pode ser criado com a tag <li>, como na exemplo abaixo:

```
<ol>
   <li>Primeiro item</li>
   <li>Segundo item</li>
   <li>Terceiro item</li>
</ol>
```

Isso produzirá uma lista numerada (que é o estilo padrão):

1. Primeiro Item
2. Segundo Item
3. Terceiro Item

Alteração manual dos números

Existem algumas maneiras pelas quais você pode jogar com os números que aparecem nos itens da lista em uma lista ordenada. O primeiro passo é definir um número inicial, usando o atributo start. A lista começará neste número definido e continuará incrementando em um como de costume.

```
<ol start="3">
   <li>Primeiro item</li>
   <li>Segundo item</li>
   <li>Terceiro item</li>
</ol>
```

Isso produzirá uma lista numerada (que é o estilo padrão):

3. Primeiro Item
4. Segundo Item
5. Terceiro Item

Você também pode definir explicitamente um determinado item da lista para um número específico. Itens de lista adicionais após um com um valor especificado continuará incrementando em um a partir do valor desse item da lista, ignorando a localização da lista pai.

```
<li value="7"></li>
```

Também é importante notar que, usando o atributo **value** diretamente em um item da lista, você pode substituir os itens de uma lista ordenada. sistema de numeração existente, reiniciando a numeração com um valor mais baixo. Portanto, se a lista de pais já estava pronta 7 e encontrou um item da lista no valor 4, esse item da lista ainda seria exibido como 4 e continuaria contando

a partir desse ponto novamente

```
<ol start="5">
   <li>Um item</li>
   <li>Outro item</li>
   <li value="4">Item de reset</li>
   <li>Outro item</li>
   <li>Mais um</li>
</ol>
```

Portanto, o exemplo acima produzirá uma lista que segue o padrão de numeração 5, 6, 4, 5, 6 - iniciando novamente em um número inferior ao anterior e duplicar o número 6 na lista.

Nota: Os atributos de início e valor aceitam apenas um número - mesmo que a lista ordenada esteja configurada para ser exibida como romana
numerais ou lette

Você pode reverter a numeração adicionando reverso no seu elemento ol:

```
<ol reversed>
   <li>Um item</li>
   <li>Outro item</li>
   <li value="4">Item de reset</li>
   <li>Outro item</li>
   <li>Mais um</li>
</ol>
```

A numeração reversa é útil se você estiver adicionando continuamente a uma lista, como em novos episódios de podcast ou apresentações e você deseja que os itens mais recentes apareçam primeiro.

Alterando o tipo de numeral

Você pode alterar facilmente o tipo de numeral mostrado no marcador de item da lista usando o atributo type

```
<ol type="1|a|A|i|I"></li>
```

Tipo	Descrição	Exemplos
1	Valor padrão - números decimais	1, 2, 3, 4
a	Ordem alfabética (minúscula)	a, b, c, d
A	Ordem alfabética (maiúscula)	A, B, C, D
i	Algarismos romanos (minúsculas)	i, ii, iii, iv
I	Algarismos romanos (maiúsculas)	I, II, III, IV

Você deve usar ol para exibir uma lista de itens, onde os itens foram intencionalmente pedidos e pedidos deve ser enfatizado. Se alterar a ordem dos itens NÃO tornar a lista incorreta, você deve use <ul>.

Lista não Ordenada

Uma lista não ordenada pode ser criada com a tag <ul> e cada item da lista pode ser criado com a tag <li>, conforme mostrado por o exemplo abaixo:

```
<ul>
   <li>Um item</li>
   <li>Outro item</li>
   <li>Mais um</li>
</ol>
```

Isso produzirá uma lista com marcadores (que é o estilo padrão):

- Um item
- Outro item
- Mais um

Você deve usar ul para exibir uma lista de itens, onde a ordem dos itens não é importante. Se alterar o Se a ordem dos itens tornar a lista incorreta, você deve usar <ol>.

Lista Aninhada

Você pode aninhar listas para representar subitens de um item da lista.

```
<ul>
   <li>item - 01</li>
   <li>Item - 02</li>
      <ul>
         <li>sub-item - 2.1</li>
         <li>sub-item - 2.2</li>
      </ul>
    <li>Item - 03 </li>
</ul>
```

- Item - 01
- Item - 02
  - sub-item - 2.1
  - sub-item - 2.2
- Item - 03

Você deve usar ul para exibir uma lista de itens, onde a ordem dos itens não é importante. Se alterar o Se a ordem dos itens tornar a lista incorreta, você deve usar <ol>.

A lista aninhada deve ser filha do elemento li. Você também pode aninhar diferentes tipos de lista:

Lista de Descrição

Uma lista de descrição (ou lista de definição, como foi chamada antes do HTML5) pode ser criada com o elemento dl. Isso consiste de grupos nome-valor, em que o nome é fornecido no elemento dt e o valor é fornecido no elemento dd.

```
<dl>
   <dt>nome - 01</dt>
   <dd>valor - 01</dd>
   <dt>nome - 02</dt>
   <dd>valor - 02</dd>
</dl>
```

Um grupo nome-valor pode ter mais de um nome e / ou mais de um valor (que representam alternativas):

```
<dl>
    <dt>nome - 01</dt>
    <dt>nome - 02</dt>
    <dd>valor - 01 e 02</dd>


     <dt>nome - 03</dt>
     <dt>valor - 03</dt>
     <dd>valor - 03</dd>
</dl>
```

Capítulo 09: Tabelas

O elemento HTML <**table**> permite que exibam dados tabulares (como texto, imagens, links, outras tabelas, etc.) em uma tabela bidimensional com linhas e colunas de células.

Tabela Simples

```
<table>
   <tr>
         <th>Título 01 / Coluna 01</th>
         <th>Título 02 / Coluna 02</th>
   </tr>
    <th>
         <td>Linha 01 / coluna 01</td>
         <td>Linha 01 / coluna 02</td>
   </th>
   <tr>
         <td>Linha 02 / coluna 01</td>
         <td>Linha 02 / coluna 02</td>
   </tr>
</table>
```

Isso renderizará uma <tabela> composta por três linhas totais (<tr>): uma linha de células de cabeçalho (<th>) e duas linhas de células de conteúdo (<td>). <th> elementos são cabeçalhos tabulares e elementos <td> são dados tabulares. Você pode colocar o que quer você quer dentro de um <td> ou <th>. Título 1 / Coluna 1 Título 2 / Coluna 2 Linha 1 coluna de dados 1 Linha 1 coluna de dados 2 Linha 2 coluna de dados 1 Linha 2 coluna de dados 2

```
<table>
   <tr>
         <td>Linha 01 / Coluna 01</td>
         <td>Linha 01 / Coluna 02</td>
         <td>Linha 01 / coluna 03</td>
   </tr>
    <tr>
         <td colspan="3">Essa segunda linha abrange as três colunas</td>
   </tr>
   <tr>
         <td rowspan="2">Linha 02 / coluna 01</td>
          <td>Linha 03 / coluna 02</td>
          <td>Linha 03 / coluna 03</td>
    </tr>
    <tr>
          <td>Linha 04 / coluna 02</td>
          <td>Linha 04 / coluna 03</td>
    </tr>
</table>
```

Veja o resultado

Linha 01 / Coluna 01	Linha 01 / Coluna 02	Linha 01 / coluna 03
Essa segunda linha abrange as três colunas		
Linha 02 / coluna 01	Linha 03 / coluna 02	Linha 03 / coluna 03
	Linha 04 / coluna 02	Linha 04 / coluna 03

Observe que você não deve criar uma tabela em que linhas e colunas se sobreponham, pois isso é torna-se um HTML inválido e o resultado é tratado de maneira diferente por diferentes navegadores da web.

rowspan = Um número inteiro não negativo que especifica o número de linhas estendidas por uma célula. O valor padrão deste atributo é um (1). Um valor zero (0) significa que a célula se estenderá da linha atual até a última linha do tabela (<**thead**>, <**tbody**> ou <**tfoot**>).

colspan = Um número inteiro não negativo que especifica o número de colunas estendidas pela célula atual. O valor padrão desse atributo é um (1). Um valor zero (0) significa que a célula se estenderá da corrente para a última coluna do grupo de colunas <grupo> em que a célula está definida.

Grupos de Colunas

Às vezes, você pode aplicar estilo a uma coluna ou grupo de colunas. Ou para fins semânticos, você pode deseja agrupar colunas. Para fazer isso, use os elementos <**colgroup**> e <**col**>.
A tag opcional <**colgroup**> permite agrupar colunas. Os elementos <**colgroup**> devem ser elementos filhos de uma <**table**> e deve vir após qualquer elemento <**caption**> e antes de qualquer conteúdo da tabela (por exemplo, <**tr**>, <**thead**>, <**corpo**> etc.).

```
<table>
    <colgroup span="2"> </colgroup>
    <colgroup span="2"> </colgroup>
</table>
```

A tag <**col**> opcional permite que você faça referência a colunas individuais ou a um intervalo de colunas sem aplicar uma lógica de agrupamento. Os elementos <**col**> são opcionais, mas, se presentes, devem estar dentro de um elemento <**colgroup**>.

```
<table>
    <colgroup>
        <col id="MinhaColunaEspecial" />
        <col />
    </colgroup>
    <colgroup>
        <col class="ColunaLegal" />
        <col class="PuraColuna" span="2" />
    </colgroup>

</table>
```

Os seguintes estilos CSS podem ser aplicados aos elementos <**colgroup**> e <**col**>:

- border;
- background;
- width;
- visibility;
- display (display: none)
  - Mostrar nenhum; removerá as colunas de exibição, fazendo com que a essas células não exista mais.

Tabelas com cabeçalho, corpo, rodapé e descrição

O HTML também fornece as tabelas com os elementos <**thead**>, <**tbody**>, <**tfoot**> e <**caption**>. Esses elementos são úteis para adicionar valor semântico às suas tabelas e fornecer um local para o estilo CSS separado.

Ao imprimir uma tabela que não cabe em uma página (papel), a maioria dos navegadores repete o conteúdo de <**thead**> na todas as páginas.

Há uma ordem específica que deve ser respeitada, e devemos estar cientes de que nem todos os elementos se encaixam como seria de esperar. O exemplo a seguir demonstra como nossos 4 elementos devem ser colocados.

```
<table>
  <caption>Título da Tabela</caption>
    <thead>
      <tr>
        <th>Cabeçalho 01</th>
        <th>Cabeçalho 02</th>
      </tr>
    </thead>

  <tbody>
      <tr>
        <td>Corpo 01</td>
        <td>Corpo 02</td>
      </tr>
  </tbody>

    <tfoot>
      <tr>
        <td>Rodapé 01</td>
        <td>Rodapé 02</td>
      </tr>
  </tfoot>
</table>
```

Os resultados do exemplo a seguir são demonstrados duas vezes - a primeira tabela não possui estilos, a segunda tabela possui alguns Propriedades CSS aplicadas: cor de fundo, cor e borda *. Os estilos são fornecidos como um guia visual e não são um aspecto essencial do tópico em questão.

Título da Tabela	
Cabeçalho 01	**Cabeçalho 02**
Corpo 01	Corpo 02
Rodapé 01	Rodapé 02

Escopo de Cabeçalho

Os elementos th são muito usados para indicar cabeçalhos para linhas e colunas da tabela, como:

```
<table>
    <thead>
        <tr>
            <td></td>
            <th>Cabeçalho 01</th>
            <th>Cabeçalho 02</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <th>Linha do Cabeçalho 01</td>
            <td></td>
            <td></td>
        </tr>
        <tr>
            <th>Linha do Cabeçalho 02</td>
            <td></td>
            <td></td>
        </tr>
    </tbody>
</table>
```

Isso pode ser aprimorado para acessibilidade usando o atributo scope. O exemplo acima seria alterado como

segue:

```
<table>
    <thead>
        <tr>
            <td></td>
            <th scope="col">Cabeçalho 01</th>
            <th scope="col">Cabeçalho 02</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <th scope="row">Linha do Cabeçalho 01</td>
            <td></td>
            <td></td>
        </tr>
    </tbody>
</table>
```

O escopo é conhecido como um atributo enumerado, o que significa que ele pode ter um valor de um conjunto específico de valores possíveis. Este conjunto inclui:

- col
- row
- colgroup
- rowgroup

Minha Referencia

Capítulo 10: Comentários

Semelhante a outras linguagens de programação, marcação e remarcação, os comentários em HTML fornecem aos desenvolvedores informações específicas de desenvolvimento sem afetar a interface do usuário. Ao contrário de outras linguagens. Este tópico explica como escrever HTML comentários e suas aplicações.

Criando comentários

Comentários em HTML podem ser usados para deixar anotações para você ou outros desenvolvedores sobre um ponto específico no código. Eles podem ser iniciado com <!- e concluído com ->, Veja o exemplo:

```
<!-- Aqui você escreve seu comentário -->
```

Eles podem ser incorporados inline em outro conteúdo:

```
<h1> HTML  <!-- Aqui você escreve seu comentário --> </h1>
```

Eles também podem abranger várias linhas para fornecer mais informações:

```
<!--
  Este é um comentário HTML de várias linhas.
  Tudo o que estiver aqui não será renderizado pelo navegador.
  Você pode "comentar" seções inteiras do código HTML.
-->
```

Capítulo 11: Classes e IDs

Atributo	Descrição
class	Indica a classe do elemento (não exclusivo)
id	Indica o ID do elemento (exclusivo no mesmo contexto)

Classes e **IDs** facilitam a referência a elementos HTML de scripts e folhas de estilo. O atributo da classe pode ser usado em uma ou mais tags e é usado pelo CSS para estilizar. Os **IDs**, no entanto, pretendem se referir a um único elemento, significando que o mesmo **ID** nunca deve ser usado duas vezes. Os **IDs** geralmente são usados com JavaScript e documento interno links e são desencorajados em CSS. Este tópico contém explicações e exemplos úteis sobre o uso adequado de atributos de classe e **ID** em **HTML**

Atribuindo classe a um elemento HTML

Classes são identificadores para os elementos aos quais estão atribuídos. Use o atributo **class** para atribuir uma classe a um elemento.

```
<div class="exemplo-classe"></div>
```

Para atribuir várias classes a um elemento, separe os nomes das classes com espaços.

```
<div class="classe1 classe2"></div>
```

Usando classes em CSS

As classes podem ser usadas para estilizar certos elementos sem alterar todos os elementos desse tipo. Por exemplo, esses dois os elementos de amplitude podem ter estilos completamente diferentes:

```
<span></span>
<span class="especial"></span>
```

Classes com o mesmo nome podem ser atribuídas a qualquer número de elementos em uma página e todos receberão o estilo associado a essa classe. Isso sempre será verdadeiro, a menos que você especifique o elemento no CSS.

Por exemplo, temos dois elementos, ambos com o destaque da classe:

```
<div class="destacar"></span>
<span class="destacar"></span>
```

Se o nosso CSS for o seguinte, a cor verde será aplicada ao texto nos dois elementos:

```
.destacar { color: green; }
```

No entanto, se queremos apenas segmentar divs com o destaque da classe, podemos adicionar especificidade como abaixo:

```
div.destacar { color: green; }
```

No entanto, ao estilizar com CSS, geralmente é recomendável que apenas classes (por exemplo, destaque) sejam usadas do que elementos com classes (por exemplo, div.highlight). Como em qualquer outro seletor, as classes podem ser aninhadas

```
.main .destacar { color: red; }
```

```
.footer > .destacar { color: blue; }
```

Você também pode encadear o seletor de classe para selecionar apenas elementos que possuem uma combinação de várias classes. Para Por exemplo, se este é o nosso HTML:

```
<div class="especial left menu"> Algum texto aqui </div>
```

E queremos colorir essa parte específica do texto em rosa, podemos fazer o seguinte em nosso CSS:

```
.especial.left.menu { color: pink}
```

Atribuindo ID a um elemento HTML

O atributo **ID** de um elemento é um identificador que deve ser ***exclusivo*** em todo o documento. Seu objetivo é identifique **exclusivamente** o elemento ao vincular (usando uma âncora), script ou estilo (com CSS).

```
<div id="exemplo-id"></div>
```

Você não deve ter dois elementos com o mesmo ID no mesmo documento, mesmo se os atributos estiverem anexados a dois tipos diferentes de elementos. Por exemplo, o seguinte código está incorreto:

```
<div id="exemplo-id"></span>
<span id="exemplo-id"></span>
```

Os navegadores farão o possível para renderizar esse código, mas poderá ocorrer um comportamento inesperado ao estilizar com CSS ou adicionar funcionalidade com JavaScript.

Para referenciar elementos por seu ID em CSS, o prefixe do ID é #

```
#exemplo-id { color: green; }
```

Para pular para um elemento com um ID em uma determinada página, adicione # com o nome do elemento na URL.

```
https://tipscode.com.br/sobre#exemplo-id
```

Esse recurso é suportado na maioria dos navegadores e não requer JavaScript ou CSS adicional para funcionar.

Valores Aceitáveis

Para um ID

Versão > 5

As únicas restrições no valor de um ID são:

1. deve ser único no documento;
2. não de conter caracteres de espaço;
3. deve conter pelo menos um caractere.

Portanto, o valor pode ter todos os dígitos, apenas um dígito, apenas caracteres de pontuação, incluir caracteres especiais, qualquer que seja. Somente sem espaço em branco.

Exemplos de valores válidos:

```
<div id="container"> ... </div>
<div id="999"> ... </div>
<div id="#%LV-||"> ... </div>
<div id="____V"> ... </div>
<div id="⌘⌥"> ... </div>
<div id="♥"> ... </div>
<div id="{}"> ... </div>
<div id="©"> ... </div>
<div id="♤₩¤☆€~¥"> ... </div>
```

Esse é invalido

```
<div id="    ">....</div>
<div id="results">....</div>
<div id="results">....</div>
```

Um valor de ID deve começar com uma letra, que pode ser seguida apenas por:

- Letras (A - Z / a -z);
- Dígitos (0 - 9);
- Hífens (" - ");
- Sublinhados (" _ ");
- Dois Pontos (" : ");
- Ponto (" . ");

Referindo-se ao primeiro grupo de exemplos na seção HTML5 acima, apenas um é válido:

```
<div id="container"> ... </div>
```

Estes também são válidos

```
<div id="textosimples"> ... </div>
<div id="texto-simples"> ... </div>
<div id="texto_simples"> ... </div>
<div id="texto:simples"> ... </div>
<div id="texto.simples"> ... </div>
```

Novamente, se não começar com uma letra (maiúscula ou minúscula), não é válido.

Para uma aula

As regras para as classes são essencialmente as mesmas que para um ID. A diferença é que os valores de classe não precisam ser exclusivo no documento.

Referindo-se aos exemplos acima, embora isso não seja válido no mesmo documento:

```
<div id="results">....</div>
<div id="results">....</div>
```

Isso também é aceito

```
<div class="results"></div>
<div class="results"></div>
```

Nota importante: Como os valores de ID e Classe são tratados fora do HTML

Lembre-se de que as regras e exemplos acima se aplicam ao contexto do HTML.
O uso de números, pontuação ou caracteres especiais no valor de uma identificação ou classe pode causar problemas em outras contextos, como CSS, JavaScript e expressões regulares. Por exemplo, embora o seguinte ID seja válido em HTML5:

```
<div id="9lions"> ... </div>
```

É inválido no CSS:

Caracteres e caso
No CSS, os identificadores (incluindo nomes de elementos, classes e IDs nos seletores) podem conter apenas os caracteres [a-zA-Z0-9] e caracteres ISO 10646 U + 00A0 e superior, mais o hífen (-) e o sublinhado (_); eles não pode começar com um dígito, dois hífens ou um hífen seguido por um dígito. (enfase adicionada) Na maioria dos casos, você pode escapar dos caracteres em contextos em que eles têm restrições ou significado especial.

Capítulo 12: Atributos de Dados

Suporte a Navegadores mais Antigos

Os atributos de dados foram introduzidos no HTML5, suportado por todos os navegadores modernos, mas navegadores mais antigos antes do HTML5 não reconhece os atributos de dados.

No entanto, nas especificações HTML, os atributos que não são reconhecidos pelo navegador devem ser deixados e o navegador simplesmente os ignorará ao renderizar a página.

Os desenvolvedores Web utilizaram esse fato para criar atributos não padrão, que são quaisquer atributos que não fazem parte das Especificações do HTML. Por exemplo, o atributo **value** na linha abaixo é considerado um atributo não padrão porque as especificações para a tag <img> não têm um atributo value e não é um atributo global:

```
<img src="exemplo.png"  value="teste"/>
```

Isso significa que, embora os atributos de dados não sejam suportados em navegadores antigos, eles ainda funcionam e você pode definir e recuperá-los usando os mesmos métodos genéricos JavaScript **setAttribute** e **getAttribute**, mas você não pode usar o nova propriedade de conjunto de dados que é suportada apenas em navegadores modernos.

Uso dos Atributos de Dados

Os atributos HTML5 data- * fornecem uma maneira conveniente de armazenar dados em elementos HTML. Os dados armazenados podem ser lidos ou modificado usando JavaScript

```
<div data-submitted="sim"  class="perfil-usuario">
    ... Conteúdo
</div>
```

- A estrutura do atributo de dados é data- *, ou seja, o nome do atributo de dados vem após a parte dos dados. Usando isto nome, o atributo pode ser acessado;
- Os dados no formato de sequência (incluindo json) podem ser armazenados usando o atributo data- *.

Capítulo 13: Vinculando Recursos

Atributos	Descrição
charset	Especifica a codificação de caracteres do documento vinculado
crossorigin	Especifica como o elemento lida com solicitações de origem cruzada
href	Especifica o local do documento vinculado
hreflang	Especifica o idioma do texto no documento vinculado
media	Especifica em qual dispositivo o documento vinculado será exibido, geralmente usado na seleção de folhas de estilo com base no dispositivo em questão
rel	Requeridos. Especifica o relacionamento entre o documento atual e o documento vinculado
ver	Especifica o relacionamento entre o documento vinculado e o documento atual.
size	Especifica o tamanho do recurso vinculado. Somente quando rel = "icon"
target	Especifica onde o documento vinculado deve ser carregado.
type	Especifica o tipo de mídia do documento vinculado
integrity	Especifica um hash codificado em base64 (sha256, sha384 ou sha512) do recurso vinculado, permitindo que o navegador para verificar sua legitimidade

Embora muitos scripts, ícones e folhas de estilo possam ser gravados diretamente na marcação HTML, é uma prática recomendada e muito mais eficiente incluir esses recursos em seu próprio arquivo e vinculá-los ao seu documento. Este tópico aborda a vinculação desses recursos externos, como folhas de estilo (CSS) e scripts (JavaScript), em um documento HTML.

JavaScript

Síncrono

```
<script src="caminho/exemplo.js" > </script>
```

A prática padrão é colocar as tags <**script**> JavaScript imediatamente antes da tag de fechamento </**body**>. Carregando seus scripts pela última vez permite que os recursos visuais do seu site sejam exibidos mais rapidamente e desencoraja seu JavaScript de tentar interagir com elementos que ainda não foram carregados.

Assíncrono

```
<script src="caminho/exemplo.js" async> </script>
```

Outra alternativa: quando o código Javascript está sendo carregado não é necessário para a inicialização da página, ele pode ser carregado de forma assíncrona, acelerando o carregamento da página. Usando a palavra reservadas **async**, o navegador carregará o conteúdo do script em paralelo e, após o download completo, interromperá a análise do HTML para analisar o arquivo Javascript.

Adiados

```
<script src="caminho/exemplo.js" defer> </script>
```

Os scripts adiados são como scripts assíncronos, com a exceção de que a análise será realizada apenas quando o HTML for totalmente analisado. É garantido que os scripts adiados sejam carregados na ordem da declaração, da mesma forma que os síncronos scripts.

<**noscript**>

```
<noscript> JavaScript disabled </noscript>
```

O elemento <**noscript**> define o conteúdo a ser exibido se o usuário tiver scripts desativados ou se o navegador não suporte usando scripts. A tag <**noscript**> pode ser colocada no <**head**> ou no <**body**>.

CSS Externo

```
<link rel="stylesheet" href="caminho.style.css" type="text/css" />
```

A prática padrão é colocar as tags CSS <**link**> dentro da tag <**head**> na parte superior do seu HTML. Desta forma, o CSS será carregado primeiro e será aplicado à sua página durante o carregamento, em vez de mostrar HTML sem estilo até que o CSS seja carregado. O atributo typeat não é necessário no HTML5, porque o HTML5 geralmente suporta CSS.

```
<link rel="stylesheet" href="caminho.style.css" type="text/css" />
```

Ou

```
<link rel="stylesheet" href="caminho.style.css"  />
```

... faça o mesmo no HTML5.

Outra prática, embora menos comum, é usar uma instrução @import dentro do CSS direto. Como isso:

```
<style type="text/css">
      @import("caminho/style.css")
</style>

<style >
      @import("caminho/style.css")
</style>
```

FavIcon

```
<link rel="stylesheet" href="caminho.style.css" type="text/css" />
```

Use o arquivo mime image / png para arquivos PNG e image / x-icon para arquivos de ícone (*.ico). Para a diferença, veja este SO questão.

Um arquivo chamado favicon.ico na raiz do seu site geralmente será carregado e aplicado automaticamente, sem a precisa de uma tag <link>. Se esse arquivo mudar, os navegadores podem ser lentos e teimosos ao atualizar seu cache.

Alternativa

```
<link rel="aternativa stylesheet"  href="caminho/estilos/style.css" title="Seu Título"  />
```

Alguns navegadores permitem a aplicação de folhas de estilo alternativas, se forem oferecidas. Por padrão, eles não serão aplicados, mas geralmente eles podem ser alterados através das configurações do navegador:

Dica de Recursos: DNS-prefetch, prefetch, prerender

Pré-Conexão

O relacionamento pré-conexão é semelhante ao dns-prefetch, na medida em que resolverá o DNS. No entanto, também tornará o Handshake TCP e negociação TLS opcional. Este é um recurso experimental.

```
<link rel="preconnect" href="URL" />
```

DNS-Prefetch

Informa os navegadores a resolver o DNS de um URL, para que todos os ativos desse URL sejam carregados mais rapidamente.

```
<link rel="dns-prefetch" href="URL" />
```

Prefetch

Informa aos navegadores que um determinado recurso deve ser pré-buscado para que possa ser carregado mais rapidamente.

```
<link rel="prefetch" href="URL" />
```

A pré-busca DNS resolve apenas o nome do domínio, enquanto a pré-busca baixa / armazena os recursos especificados.

Prerender

Informa os navegadores a buscar e renderizar a URL em segundo plano, para que possam ser entregues ao usuário instantaneamente enquanto o usuário navega para esse URL. Este é um recurso experimental.

```
<link rel="prerender" href="URL" />
```

Atributo media

```
<link rel="stylesheet" href="test.css" media="print" />
```

Mídia especifica qual folha de estilo deve ser usada para que tipo de mídia. O uso do valor de impressão seria exibido apenas essa folha de estilo para as páginas impressas.

O valor desse atributo pode ser qualquer um dos valores de tipo de mídia (semelhante a uma consulta de mídia CSS).

Prev e Next

Quando uma página faz parte de uma série de artigos, por exemplo, é possível usar prev e next para apontar para as páginas que estão chegando antes e depois.

```
<link rel="prev"  href="https://www.tipscode.com.br" />
```

```
<link rel="next"  href="https://www.tipscode.com.br" />
```

Capítulo 14: Incluindo JavaScript no HTML

www.tipscode.com.br | Alisson Suassuna

Atributos	Descrições
src	Especifica o caminho para um arquivo JavaScript. Uma URL relativa ou absoluta
type	Especifica o tipo **MIME**. Este atributo é obrigatório em HTML4, mas opcional em HTML5
async	Especifica que o script deve ser executado de forma assíncrona (apenas para scripts externos). Este atributo não requer nenhum valor (exceto em XHTML).
defer	Especifica que o script deve ser executado quando a página terminar de analisar (apenas para scripts). Este atributo não requer nenhum valor (exceto em XHTML).
charset	Especifica a codificação de caracteres usada em um arquivo de script externo, por exemplo UTF-8
crossorigin	Como o elemento lida com solicitações de origens cruzadas
nonce	O nonce criptográfico usado na política de segurança de conteúdo verifica o CSP3

Manipulação de JavaScript Desativado

É possível que o navegador do cliente não suporte Javascript ou tenha a execução Javascript desativada, talvez devido por razões de segurança. Para poder informar aos usuários que um script deve ser executado na página, a tag <noscript> pode ser usado. O conteúdo de <noscript> é exibido sempre que o Javascript estiver desativado para a página atual.

```
<script type="text/css">
      document.write("Olá, Mundo!")
</script>
<noscript> Este navegador não suporta Javascript. </noscript>
```

Vinculando a um Arquivo JavaScript Externo

```
<script src="exemplo.js"> </script>
```

O atributo **src** funciona como o atributo **href** em em um contexto de âncoras: você pode especificar um URL absoluto ou relativo. O exemplo acima é vinculado a um arquivo dentro do mesmo diretório do documento HTML. Isso geralmente é adicionado dentro do Tags <**head**> na parte superior do documento html

Vinculando Arquivo Diretamente

Em vez de vincular a um arquivo externo, você também pode incluir o código JS como está no seu HTML:

```
<script >
  // Código JavaScript
</script>
```

Vinculando Arquivo Javascript em execução Assíncrona

```
<script type="text/javascript" src="URL"> </script>
```

Capítulo 15: Usando HTML com CSS

CSS fornece estilos para elementos HTML na página. O estilo embutido envolve o uso do atributo **style** em tags e é altamente desencorajado. As folhas de estilo internas usam a tag <**style**> e são usadas para declarar regras para partes direcionadas de uma página. Folhas de estilo externas podem ser usadas por meio de uma tag <**link**> que pega um arquivo externo de CSS e aplica as regras para o documento. Este tópico aborda o uso dos três métodos de conexão.

Usando CSS externo

Use o atributo link no cabeçalho do documento:

```
<head>
    <link rel="stylesheet" type="text/css" href="style.css" />
</head>
```

Você também pode usar folhas de estilo fornecidas em sites por meio de uma rede de entrega de conteúdo ou CDN, para abreviar. (por exemplo, Bootstrap):

```
<head>
    <link rel="stylesheet"
 href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
</head>
```

Geralmente, você pode encontrar o suporte da CDN para uma estrutura em seu site.

CSS Interno

Você também pode incluir elementos CSS internamente usando a tag <style>

```
<head>
    <style type="text/css">
        body {
            backgroud-color: red;
        }
    </style>
</head>
```

CSS Inline

Você pode estilizar um elemento específico usando o atributo **style**

```
<span style="color: red"> Esse texto terá a cor vermelha </span>
```

Nota: Tente evitar isso - Ás boas pratica diz que temos que separar os arquivos CSS dos arquivos HTML.

Multiplas chamadas CSS

```
<head>
    <link rel="stylesheet" type="text/css" href="style1.css" />
    <link rel="stylesheet" type="text/css" href="style2.css" />
</head>
```

Observe que arquivos e declarações posteriores substituirão os anteriores. Portanto, se o arquivo style1.css contiver:

```
<head>
    <style type="text/css">
        body {
            backgroud-color: red;
        }
    </style>
</head>
```

e style2.css contém:

```
<head>
    <style type="text/css">
        body {
            backgroud-color: blue;
        }
    </style>
</head>
```

se os dois forem usados, o plano de fundo do documento será azul.

Capítulo 16: Imagens

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
src	Especifica a URL da imagem
srcset	Imagens para usar em diferentes situações (por exemplo, telas de alta resolução, monitores pequenos etc.)
sizes	Tamanhos de imagem entre pontos de interrupção
crossorigin	Como o elemento lida com solicitações de origens cruzadas
usemap	Nome do mapa da imagem a ser usado
ismap	Se a imagem é um mapa de imagens do servidor
alt	Texto alternativo que deve ser exibido se, por algum motivo, a imagem não puder ser exibida
width	Especifica a largura da imagem (opcional)
height	Especifica a altura da imagem (opcional)

Criando Imagem

Para adicionar uma imagem a uma página, use a tag <**img**>.

A Tag de imagem (img) não possuem uma tag de fechamento. Os dois principais atributos que você atribui à tag img são **src**, a fonte da imagem e **alt**, que é um texto alternativo que descreve a imagem.

```
<img src="imagens/alisson.png" alt="Foto do Alisson Suassuna" />
```

Você também pode obter imagens de um URL da web:

```
<img src="https://tipscode.com.br/alisson.png" alt="Foto do Alisson Suassuna do site" />
```

Nota: As imagens não são tecnicamente inseridas em uma página HTML, as imagens são vinculadas às páginas HTML. A tag <img> cria um espaço de espera para a imagem referenciada.

Também é possível incorporar imagens diretamente dentro da página usando base64:

```
<img src="data:imagem/png;base64,iVBOR..." alt="Olá, Mundo" />
```

Texto alternativo

O texto alternativo é usado pelos leitores de tela para usuários com deficiência visual e pelos mecanismos de pesquisa. Portanto, é importante escreva um bom texto alternativo para suas imagens.

Imagens responsivas usando o atributo srcset

Usando srcset com tamanhos

```
<img sizes="(min-width: 1200px) 580px, (min-width: 640px) 48vw, 98vw"
srcset="img/alisson-300.png 300w, img/alisson-600.png 600w, img/alisson-900.png
900w, imag/alisson-1200.png 1200w" src="img/alisson-900.png" alt="Foto Alissom" />
```

tamanhos são como consultas de mídia, descrevendo quanto espaço a imagem ocupa da janela de exibição.

- Se a **viewport** for maior que 1200 px, a imagem terá exatamente 580 px (por exemplo, nosso conteúdo será centralizado no contêiner com uma largura máxima de 1200 px. A imagem ocupa metade dela menos margens);
- Se a **viewport** for maior que 1200 px, a imagem terá exatamente 580 px (por exemplo, nosso conteúdo será centralizado no contêiner com uma largura máxima de 1200 px. A imagem ocupa metade dela menos margens);
- Se a **viewport** tiver outro tamanho, no nosso caso, menor que 640 px, a imagem ocupará 98% da **viewport** (por exemplo, image é dimensionado com a nossa página e ocupa a largura total da janela de visualização menos margens). A condição da mídia deve ser omitida para último item.

O **srcset** está apenas informando ao navegador quais imagens temos disponíveis e quais são seus tamanhos.

- img/alisson-300.jpg is 300px largura;
- img/alisson-600.jpg is 600px largura;
- img/alisson-900.jpg is 900px largura;
- img/alisson-1200.jpg is 1200px largura.

src é sempre fonte de imagem obrigatória. No caso de usar com srcset, o src servirá a imagem de fallback caso o navegador seja não suporta srcset.

Usando srcset sem tamanhos

```
<img src="img/alisson-300.png" alt="Foto do Alisson" srcset="img/alisson-300.png 1x,
img/alisson-600.png 2x, img/alisson-1200 3x" />
```

srcset fornece uma lista de imagens disponíveis, com proporção de pixel de dispositivo x descritor.

- Se a proporção dispositivo-pixel for 1, use img / alisson-300.png;
- Se a proporção dispositivo-pixel for 2, use img / alisson-600.png;
- Se a proporção dispositivo-pixel for 3, use img / alisson-900.png

src é sempre fonte de imagem obrigatória. No caso de usar com srcset, o src servirá a imagem de fallback caso o navegador seja não suporta **srcset**.

Imagem responsiva com o elemento <picture>

```
<picture>
    <source media="(min-width: 600px)" srcset="imagemGrande.jpg" />
    <source media="(min-width: 450px)" srcset="imagemPequena.jpg" />
    <img src="imagemPadrao.jpg" style="width: auto;"  />
</picture>
```

Uso

Para exibir imagens diferentes com largura de tela diferente, você deve incluir todas as imagens usando a tag de origem em um etiqueta de imagem, como mostrado no exemplo acima.

Resultado

- Em telas com largura de tela> 600px, mostra imagemGrande.jpg;
- Em telas com largura de tela> 450px, mostra imagemPequena.jpg;
- Em telas com outra largura de tela, mostra imagemPadrao.jpg

Capítulo 17: Mapas de Imagens

www.tipscode.com.br | Alisson Suassuna	
Tags / Atributos	Descrições
<img>	Abaixo estão os atributos específicos do mapa de imagem para usar com <img>. Atributos <img> regulares se aplicam.
usemap	O nome do mapa com um símbolo de hash anexado a ele. Por exemplo, para um mapa com nome = "mapa", a imagem deve ter usemap = "# map"
<map>	
name	O nome do mapa para identificá-lo. Para ser usado com o atributo usemap da imagem.
<area>	Abaixo estão os atributos específicos da <área>. Quando href é especificado, tornando a <área> um link, a <área> também suporta todos os atributos da marca âncora (<a>), exceto o ping
alt	O texto alternativo a ser exibido se as imagens não forem suportadas. Isso só é necessário se href também estiver definido em a <área>.
coords	As coordenadas que descrevem a área selecionável. Quando shape = "polygon", isso deve ser definido como uma lista de pares "x, y" separados por vírgulas (isto é, shape = "polygon" coords = "x1, y1, x2, y2, x3, y3, ..."). Quando shape = "rectangle", deve ser definido para a esquerda, superior, direita, inferior. Quando shape = "circle", isso deve ser definido como centerX, centerY, raio.
href	A URL do hiperlink, se especificado. Se for omitido, a <área> não representará um hiperlink
shape	A forma da <área>. Pode ser definido como padrão para selecionar a imagem inteira (nenhum atributo de cordas) necessário), círculo ou circ para um círculo, retângulo ou retângulo para um retângulo e polígono ou poli para um área poligonal especificada por pontos de canto.

Introdução a mapa de imagem

Descrição

Um mapa de imagem é uma imagem com áreas clicáveis que geralmente atuam como hiperlinks. A imagem é definida pela tag <**img**> e o mapa é definido por uma tag <**map**> com a tag <**area**> para indicar cada área clicável. Use os atributos **usemap** e **name** para vincular a imagem e o mapa.

Exemplo básico

Para criar um mapa de imagem para que cada uma das formas da imagem abaixo seja clicável:

O código seria o seguinte:

```
<img src="https://tipscode.com.br/imgs/alisson.gif" usemap="#shapes" />
<map name="shapes"
      <area shape="polygon"  coords="79,6,5 134, 153, 134" />
      <area shape="rectangle"  coords="177,6,306,134" />
      <area shape="circle"  coords="397,71,65" />
</map>
```

Capítulo 18: Elementos de controle de Entrada

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
class	Indica a classe de entrada
id	Indica o ID de entrada
type	Identifica o tipo de controle de entrada a ser exibido. Os valores aceitáveis estão ocultos, texto, tel, URL, email, senha, data, hora, número, intervalo, cor, caixa de seleção, rádio, arquivo, envio, imagem, redefinição e botão. O padrão é texto, se não especificado, se o valor for inválido ou se o navegador não suportar o tipo Especificadas.
name	Indica o nome da entrada
disabled	O valor booleano que indica a entrada deve ser desativado. Os controles desativados não podem ser editados, são não é enviado no envio do formulário e não pode receber o foco.
checked	Quando o valor do atributo type é radio ou checkbox, a presença desse atributo booleano indica que o controle está selecionado por padrão; caso contrário, será ignorado.
multiple	HTML5 Indica que vários arquivos ou valores podem ser transmitidos (aplica-se apenas a entradas de arquivo e tipo de email)
placeholder	HTML5 Uma dica para o usuário sobre o que pode ser inserido no controle. O texto do espaço reservado não deve
autorocomplete	HTML5 Indica se o valor do controle pode ser preenchido automaticamente pelo navegador.
readonly	Valor booleano que indica que a entrada não é editável. Os controles somente leitura ainda são enviados no formulário envio, mas não receberá o foco. HTML5: este atributo é ignorado quando o valor do tipo O atributo é definido como oculto, intervalo, cor, caixa de seleção, rádio, arquivo ou botão.
required	HTML5 Indica que um valor deve estar presente ou o elemento deve ser verificado para que o formulário seja seja submetido
alt	Um texto alternativo para imagens, caso elas não sejam exibidas.
autofocus	O elemento <input> deve ter o foco quando a página é carregada.
value	Especifica o valor do elemento <input>.
step	O atributo step especifica os intervalos de número legal. Funciona com os seguintes tipos de entrada: número, intervalo, data, data e hora local, mês, hora e semana.

Um componente-chave dos sistemas interativos da web são as tags de entrada que são elementos HTML projetados para assumir uma forma específica de entrada de usuário. Diferentes tipos de elementos de entrada podem regular os dados inseridos para se ajustarem a um formato especificado e fornecer segurança à entrada de senha.

Texto (Text)

O tipo de entrada mais básico e a entrada padrão, se nenhum tipo for especificado. Esse tipo de entrada define um campo de texto de linha única com quebras de linha removidas automaticamente do valor de entrada. Todos os outros caracteres podem ser inseridos nisso. <**input**> elementos são usados dentro de um elemento <**form**> para declarar controles de entrada que permitem que os usuários insiram dados.

Sintaxe

```
<input type="text" />
```

ou (sem especificar um tipo, usando o atributo padrão):

```
<input />
```

A largura padrão de uma entrada de campo de texto é de 20 caracteres. Isso pode ser alterado especificando um valor para o tamanho atributo como este:

```
<input type="text" size="50" />
```

O atributo **size** é muito diferente de definir uma largura com CSS. O uso de uma largura define um valor específico (em número de pixels, porcentagem do elemento pai etc.) que a entrada deve sempre ser ampla. Usando o tamanho calcula a quantidade de largura a ser alocada com base na fonte que está sendo usada e na largura dos caracteres normalmente.

Nota: O uso do atributo size não limita inerentemente o número de caracteres que podem ser inseridos no caixa, apenas a largura da caixa é exibida. Para limitar o comprimento, consulte Validação de entrada.

Um campo de entrada permite apenas uma linha de texto. Se você precisar de uma entrada de texto de várias linhas para uma quantidade substancial de texto, use um

Botões com atributos Checkbox e Radio

Visão geral

As caixas de seleção e os botões de opção são gravados com a tag HTML <input>, e seu comportamento é definido no HTML especificação.

A caixa de seleção ou botão de opção mais simples é um elemento <input> com um atributo de tipo de caixa de seleção ou opção, respectivamente:

```
<input type="checkbox" />
<input type="radio" />
```

Um único elemento de caixa de seleção independente é usado para uma única opção binária, como uma pergunta de sim ou não. Caixas de seleção são independentes, o que significa que o usuário pode selecionar quantas opções desejar em um grupo de caixas de seleção. No Em outras palavras, marcar uma caixa de seleção não desmarca as outras caixas no grupo de caixas de seleção.

Os botões de opção geralmente vêm em grupos (se não estiver agrupado com outro botão de opção, você provavelmente pretendia usar um em vez disso) identificada usando o mesmo atributo de nome em todos os botões desse grupo. A seleção dos botões de opção são mutuamente exclusivos, o que significa que o usuário pode selecionar apenas uma opção em um grupo de botões de opção. Quando um botão de opção é marcado, qualquer outro botão de opção com o mesmo nome que foi verificado anteriormente se torna desmarcada.

Exemplo:

```
<input type="radio" name="color" id="red" value="#Foo" />
<input type="radio" name="color" id="green" value="#oFo" />
<input type="radio" name="color" id="blue" value="#ooF" />
```

Quando visualizados, os botões de opção aparecem como um círculo (desmarcado) ou um círculo preenchido (marcado). As caixas de seleção aparecem como quadrado (desmarcado) ou um quadrado preenchido (marcado). Dependendo do navegador e do sistema operacional, o quadrado às vezes tem cantos arredondados.

Atributos

caixas de seleção e botões de opção têm vários atributos para controlar seu comportamento:

valor

Como qualquer outro elemento de entrada, o atributo **value** especifica o valor da sequência a ser associada ao botão no evento de envio do formulário. No entanto, caixas de seleção e botões de opção são especiais, pois quando o valor é omitido, o padrão é ativado quando enviado, em vez de enviar um valor em branco. O atributo value não é refletido no aparência do botão.

O atributo marcado especifica o estado inicial de uma caixa de seleção ou botão de opção. Este é um atributo booleano e pode ser omitido. Cada uma delas é válida, maneiras equivalentes para definir um botão de opção marcado:

```
<input checked />
<input checked="" />
<input checked="checked" />
<input checked="ChEcKeD" />
```

A ausência do atributo verificado é a única sintaxe válida para um botão desmarcado:

```
<input type="checkbox" />
<input type="radio" />
```

Ao redefinir um <**form**>, as caixas de seleção e os botões de opção são revertidos para o estado de seu atributo marcado.

Acessibilidade
Labels

Para contextualizar os botões e mostrar aos usuários a finalidade de cada botão, cada um deles deve ter um rótulo. Isso pode ser feito usando um elemento <**label**> para quebrar o botão. Além disso, isso torna o rótulo clicável, para que você selecione o botão correspondente.

Exemplo:

```
<label>
   <input type="radio" name="color" value="#Foo"  />
    Vermelho
</label>
```

ou com um elemento <**label**> com um atributo for definido como o atributo id do botão:

```
<input type="radio" name="color" value="#Foo"  id="red" />
<label for="red">Vermelho...</label>
```

Grupos de botões

Como cada botão de opção afeta os outros do grupo, é comum fornecer um rótulo ou contexto para todo o grupo de botões de opção. Para fornecer um rótulo para todo o grupo, os botões de opção devem ser incluídos em um elemento <**fieldset**> com um elemento <**legend**> dentro dele.

Exemplo:

```
<fieldset>
     <legend>Temas de Cores:</legend>
     <p>
        <input type="radio" name="color" id="red" value="#Foo" />
        <label for="vermelho"> Vermelho.. </label>
    </p>
    <p>
      <input type="radio" name="color" id="green" value="#oFo" />
      <label for="green"> Verde.. </label>
    </p>
     <p>
       <input type="radio" name="color" id="blue" value="#ooF" />
       <label for="blue"> Azul.. </label>
     </p>
</fieldset>
```

As caixas de seleção também podem ser agrupadas de maneira semelhante, com um conjunto de campos e uma legenda identificando o grupo de caixas de seleção. No entanto, lembre-se de que as caixas de seleção não devem compartilhar o mesmo nome porque não são mutuamente exclusivos. Isso fará com que o formulário envie vários valores para a mesma chave e nem todos os idiomas do servidor lidam com isso da mesma maneira (comportamento indefinido).

Cada caixa de seleção deve ter um único nome ou use um conjunto de colchetes ([]) para indicar que o formulário deve enviar uma matriz de valores para essa chave.

O método escolhido deve depender de como você planeja manipular os dados do formulário no lado do cliente ou no servidor. Vocês também deve manter a legenda curta, pois algumas combinações de navegadores e leitores de tela leem a legenda antes de cada campo de entrada no conjunto de campos

Entradas Validadas (input)

A validação de entrada HTML é feita automaticamente pelo navegador, com base em atributos especiais no elemento de entrada. isto pode substituir parcial ou completamente a validação de entrada JavaScript. Esse tipo de validação pode ser contornado pelo usuário por meio de solicitações HTTP especialmente criadas, para que não substitua a validação de entrada do servidor. Apenas a validação ocorre ao tentar enviar o formulário, portanto, todas as entradas restritas devem estar dentro de um formulário para que a validação ocorrer (a menos que você esteja usando JavaScript). Lembre-se de que entradas desabilitadas ou somente leitura não acionarão validação.

Alguns tipos de entrada mais recentes (como **email**, **url**, **tel**, **data** e muito mais) são validados automaticamente e não requerem suas próprias restrições de validação.

Required

Use o atributo necessário para indicar que um campo deve ser preenchido para passar na validação.

```
<input required />
```

Comprimento mínimo / máximo

Use os atributos **minlength** e **maxlength** para indicar os requisitos de comprimento. A maioria dos navegadores impedirá o usuário de digitar mais de um máximo de caracteres na caixa, impedindo-os de invalidar sua entrada mesmo antes eles tentam enviar.

```
<input minlength="3"  />
<input minlength="15" />
<input minlength="3  manlength="15" />
```

Especificando um intervalo

Use atributos mínimo e máximo para restringir o intervalo de números que um usuário pode inserir em um número ou tipo de tipo

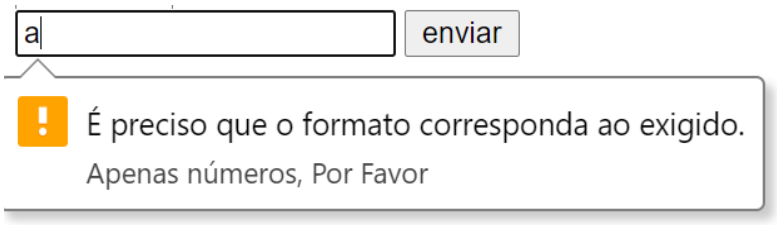
```
Marcas: <input type="number" size="6" name="marcas" min="0" max="100" />
Assunto: <input type="range" size="2" name="assunto" min="1" max="5" />
```

Corresponder a um padrão

Para obter mais controle, use o atributo padrão para especificar qualquer expressão regular que deve ser correspondida para passe na validação. Você também pode especificar um título, incluído na mensagem de validação, se o campo não for aprovado.

```
<input pattern="\id*" title="Apenas números, por favor"  />
```

Aqui está a mensagem mostrada no Google Chrome versão 51 ao tentar enviar o formulário com um valor inválido dentro deste campo:



Nem todos os navegadores exibem uma mensagem para padrões inválidos, embora exista suporte total entre os aplicativos modernos mais usados. navegadores.

Verifique o suporte mais recente no Can I Use e implemente de acordo.

Aceitar tipo de arquivo

Para campos de entrada do tipo arquivo, é possível aceitar apenas certos tipos de arquivos, como vídeos, imagens, áudios, extensões de arquivo específicas ou determinados tipos de mídia. Por exemplo:

```
<input type="file" accept="imagem/*"  title="Somente imagens são permitidas" />
```

Vários valores podem ser especificados com vírgula, por exemplo:

```
<input type="file" accept="imagem/*, .rar,aplicacao/zip" />
```

Nota: A adição do atributo novalidate ao elemento do formulário ou do atributo formnovalidate ao botão de envio evita validação em elementos de formulário. Por exemplo:

```
<form>
        <input type="text" name="name" required />
        <input type="email" name="email" required />
        <input pattern="\d*" name="number" required />

        <input type="submit" value="Publish" />
        <input type="submit" value="Save" formovalidate />
</form>
```

O formulário possui campos necessários para "publicar" o rascunho, mas não são necessários para "salvar" o rascunho

Cores

```
<input type="color" name="favcolor" value="#ffooo" />
```

Nos navegadores de suporte, o elemento de entrada com um atributo de tipo cujo valor é cor cria um controle semelhante a um botão, com uma cor igual ao valor do atributo de cor (o padrão é preto se o valor não for especificado ou for inválido formato hexadecimal).

Clicar neste botão abre o widget de cores do sistema operacional, que permite ao usuário selecionar uma cor.

O fallback para navegadores que não suportam esse tipo de entrada é um tipo de entrada regular = texto.

#ff0000

Senha

```
<input type="password" name="senha"  />
```

O elemento de entrada com um atributo de tipo cujo valor é senha cria um campo de texto de linha única semelhante à entrada type = text, exceto que o texto não é exibido quando o usuário o digita.

```
<input type="password" name="senha" placeholder="Senha" />
```

O texto do placeholder é mostrado em texto sem formatação e substituído automaticamente quando um usuário começa a digitar.

Nota: Alguns navegadores e sistemas modificam o comportamento padrão do campo de senha para exibir também os caractere digitado recentemente por um curto período, da seguinte forma:

Arquivos (File)

```
<input type="file" name="fileSubmission" />
```

As entradas de arquivo permitem que os usuários selecionem um arquivo de seu sistema de arquivos local para uso com a página atual. Se usado em conjunto com um elemento de formulário, eles podem ser usados para permitir que os usuários enviem arquivos para um servidor (para obter mais informações, consulte Upload de arquivos). O exemplo a seguir permite que os usuários usem a entrada de arquivo para selecionar um arquivo do sistema de arquivos e fazer upload desse arquivo para um script no servidor chamado upload_file.php.

```
<form action="upload_file.php" method="post" enctype="multipart/form-data">
    Seleção de arquivos para enviar:
    <input type="file" name="fileSubmission" id="fileSubmission"/>
    <input type="email" value="Arquivo enviado" name="submit" />
</form>
```

Vários arquivos

Adicionando o atributo múltiplo, o usuário poderá selecionar mais de um arquivo

```
<input type="file" name="fileSubmission" id="fileSubmission" multiple />
```

Aceitar arquivos

O atributo Accept especifica os tipos de arquivos que o usuário pode selecionar. Por exemplo. .png, .gif, .jpeg.

```
<input type="file" name="fileSubmission" accept="image/x-png,image/gif" />
```

Botões

```
<input type="button" value="texto do botão"  />
```

Os botões podem ser usados para acionar ações que ocorrem na página, sem enviar o formulário. Você também pode usar o <**button**>, se você precisar de um botão que possa ser estilizado com mais facilidade ou que contenha outros elementos:

```
<button type="button">Texto do Botão</button>
```

Os botões são normalmente usados com um evento "onclick":

```
<input type="button" onclick="alert("Olá, Mundo")" value="Clique Aqui" />
```

Ou

```
<button type="button" onclick="('Olá, Mundo')">Clique aqui</button>
```

Atributos
[name]

O nome do botão, enviado com os dados do formulário.

[type]

O tipo do botão.

Os valores possíveis são:
enviar: o botão envia os dados do formulário ao servidor. Esse é o padrão se o atributo não for especificado ou se o atributo atributo é alterado dinamicamente para um valor vazio ou inválido.

reset: O botão redefine todos os controles para seus valores iniciais.

botão: o botão não possui comportamento padrão. Pode ter scripts do lado do cliente associados aos eventos do elemento, que são acionados quando os eventos ocorrem.

menu: o botão abre um menu pop-up definido por meio do elemento designado.

[value]

O valor inicial do botão.

www.tipscode.com.br | Alisson Suassuna

Atributos	Descrições
form	Especifica o ID do formulário ao qual o botão pertence. Se nenhum for especificado, ele pertencerá ao seu elemento de formulário ancestral (se houver).
formaction	Especifica para onde enviar os dados do formulário. quando o formulário é enviado usando este botão.
formenctype	Especifica como os dados do formulário devem ser codificados. ao enviá-lo ao servidor usando este botão. Só pode ser usado com formmethod = "post".
formmethod	Especifica o método HTTP a ser usado (POST ou GET) ao enviar dados do formulário usando este botão.
formovalidate	e Especifica que os dados do formulário não devem ser validados no envio
formtarget	Specifies where to display the response that is received after submitting the form using this button.

Enviar (Submit)

```
<input type="submit" value="submit" />
```

Uma entrada de envio cria um botão que envia o formulário que está dentro quando clicado. Você também pode usar o elemento <**button**> se precisar de um botão de envio que possa ser estilizado com mais facilidade ou conter outros elementos:

```
<button type="submit">
    <img src="submit-icon.jpg" /> Enviar
</button>
```

Redefinir(Reset)

```
<input type="reset" value="Redefinir" />
```

Uma entrada do tipo reset cria um botão que, quando clicado, redefine todas as entradas no formulário em que estão contidas. Estado padrão.

- O texto em um campo de entrada será redefinido para em branco ou seu valor padrão (especificado usando o atributo value).
- Quaisquer opções no menu de seleção serão desmarcadas, a menos que tenham o atributo selecionado.
- Todas as caixas de seleção e caixas de opção serão desmarcadas, a menos que tenham o atributo marcado.

Nota: Um botão de redefinição deve estar dentro ou anexado a (através do atributo do formulário) um elemento <**form**> para ter qualquer efeito. O botão redefinirá apenas os elementos neste formulário.

Entrada Oculta (Hidden)

```
<input type="hidden" name="NomeDoInput" value="valorInput" />
```

O atributo hidden não ficará visível para o usuário, mas seu valor será enviado ao servidor quando o formulário for enviado Não obstante.

Telefone (Tel)

```
<input type="tel" value="559999999" />
```

O elemento de entrada com um atributo de tipo cujo valor é **tel** representa um controle de edição de texto simples de uma linha para inserir um número de telefone.

Email

O **<input type = "email">** é usado para campos de entrada que devem conter um endereço de email.

```
<form>
    <label> E-mail: </label>
    <input type="email" name="email" />
</form>
```

O endereço de e-mail pode ser validado automaticamente quando enviado, dependendo do suporte do navegador.

Números (Number)

```
<input type="number" value="0" name="quantity" />
```

O elemento Input com um atributo type cujo valor é number representa um controle preciso para definir as propriedades do elemento valor para uma sequência que representa um número.

Observe que este campo não garante um número correto. Apenas permite todos os símbolos que poderiam ser usado em qualquer número real, por exemplo, o usuário poderá inserir valores como e1e-, 0.

Intervalo (Range)

```
<input type="range" min=" " max=" " step=" " />
```

Um controle para inserir um número cujo valor exato não é importante.

Atributo	Descrição	Valor Padrão
min	Valor mínimo para o intervalo	0
max	Valor máximo para o intervalo	100
step	Valor a aumentar em cada incremento	1

Pesquisar (Search)

A pesquisa por tipo de entrada é usada para pesquisa textual. Ele adicionará um símbolo de lupa ao lado do espaço para texto na maioria dos navegadores

```
<input type="search" value="googlesearch" />
```

Imagem

```
<input type="image" src="img.png" alt="nome da imagem" height="50px" width="50px" />
```

Uma imagem. Você deve usar o atributo src para definir a origem da imagem e o atributo alt para definir. Texto Alternativo. Você pode usar os atributos de altura e largura para definir o tamanho da imagem em pixels.

Dia da semana (Week)

```
<input type="week" />
```

Dependendo do suporte do navegador, será exibido um controle para inserir um número de semana e um número de semana sem fuso horário.

Url

```
<input type="week"  name="Homepage"/>
```

Isso é usado para campos de entrada que devem conter um endereço de URL.

Dependendo do suporte do navegador, o campo de URL pode ser validado automaticamente quando enviado.

Alguns smartphones reconhecem o tipo de URL e adicionam ".com" ao teclado para corresponder à entrada de URL.

DateTime-Local

```
<input type="datetime-local" />
```

Dependendo do suporte do navegador, um seletor de data e hora será exibido na tela para você escolher uma data e hora.

Mês (Month)

```
<input type="month" />
```

Dependendo do suporte do navegador, um controle será exibido para escolher o mês.

Tempo (Time)

```
<input type="time" />
```

A entrada de hora marca esse elemento como aceitando uma sequência que representa uma hora. O formato é definido na RFC 3339 e deve ser um período parcial, como

20:05:67
10:12:24.09

Atualmente, todas as versões do Edge, Chrome, Opera e Chrome para Android suportam o tipo = "time". As versões mais recentes do navegador Android, especificamente 4.4 e superior. O Safari para iOS oferece suporte parcial, sem suporte mínimo, máximo, e atributos de etapa.

Data e horas (DateTime (Global))

O elemento de entrada com um atributo de tipo cujo valor é "datetime" representa um controle para definir as propriedades do elemento valor para uma sequência que representa uma data e hora globais (com informações de fuso horário).

```
<fieldset>
    <p> <label>Hora<input type="datetime" name="hora" /> </label></p>
</fieldset>
```

Atributos permitidos:

- global attributes
- name
- disabled
- form
- type
- autocomplete
- autofocus
- list
- min & max
- step (float)
- readonly
- required value

Data (Date)

```
<input type="date" />
```

Um seletor de datas será exibido na tela para você escolher uma data. Isso não é suportado no Firefox ou no Internet Explorer.

Aprenda JavaScript criando um Jogo do ZERO

Você vai aprender

- Manipulação DOM
- Criação de Elementos com JavaScript;
- Animação de Personagem com JavaScript
- Inserção de Elemenos Externos.
- Criação de Layoout com CSS3

Inscreve-se

Capítulo 19: Formulários HTML

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
accept-charset	Especifica as codificações de caracteres que devem ser usadas para o envio do formulário.
action	Especifica para onde enviar os dados do formulário quando um formulário é enviado
autocomplete	Especifica se um formulário deve ter o preenchimento automático ativado ou desativado.
enctype	Especifica como os dados do formulário devem ser codificados ao enviá-los ao servidor (apenas para method = "post").
method	Especifica o método HTTP a ser usado ao enviar dados do formulário (POST ou GET).
name	Especifica o nome de um formulário
novalidate	Especifica que o formulário não deve ser validado quando enviado.
target	Especifica onde exibir a resposta que é recebida após o envio do formulário.

Para agrupar elementos de entrada e enviar dados, o HTML usa um elemento de formulário para encapsular entrada e envio de elementos. Esses formulários tratam do envio dos dados no método especificado para uma página manipulada por um servidor ou manipulador. Este tópico explica e demonstra o uso de formulários HTML na coleta e envio de dados de entrada.

Enviando (Submitting)

Atributo Action (Ação)

O atributo **action** define a ação a ser executada quando o formulário é enviado, o que geralmente leva a um script que coleta as informações enviadas e trabalha com elas. se você deixar em branco, ele será enviado para o mesmo arquivo

```
<form action="acao.js" />
```

Atributo method (método)

O atributo method é usado para definir o método HTTP do formulário que é GET ou POST

```
<form action="acao.js" method="get" >
<form action="acao.js" method="post" >
```

O método GET é usado principalmente para obter dados, por exemplo, para receber uma postagem por seu ID ou nome ou para enviar uma pesquisa inquerir. O método GET anexará os dados do formulário à URL especificada no atributo action.

```
www.exemplo.com/acao.js?nome=alisson&sobrenome=suassuna
```

O método POST é usado ao enviar dados para um script. O método POST não anexa os dados do formulário ao o URL da ação, mas envia usando o corpo da solicitação.

Para enviar os dados do formulário corretamente, um nome de atributo de nome deve ser especificado. Como exemplo, vamos enviar o valor do campo e definir seu nome como sobrenome:

```
<input type="acao.js" name="sobrenome" value="Suassuna"  />
```

Mais atributos

```
<form action="acao.js" method="post"  target="_blank" accept-charset="UTF-8"
enctype="application/x-www-form-urlencode" autocomplete="off" novalidate>
</form>
```

Atributo target da tag <form>

O atributo target especifica um nome ou uma palavra-chave que indica onde exibir a resposta recebida depois de enviar o formulário.

O atributo de destino define um nome ou uma palavra-chave para um contexto de navegação (por exemplo, guia, janela ou quadro embutido).

De Tag com um atributo de destino:

```
<form tarrget="_blank">
```

Atributo	Descrição
_blank	A resposta é exibida em uma nova janela ou guia
_self	A resposta é exibida no mesmo quadro (este é o padrão)
_parent	A resposta é exibida no quadro pai
_top	A resposta é exibida no corpo inteiro da janela
framename	nome da estrutura A resposta é exibida em um iframe nomeado

Nota: O atributo de destino foi descontinuado no HTML 4.01. O atributo de destino é suportado em HTML5.

Quadros e conjuntos de quadros não são suportados no HTML5; portanto, os valores _parent, _top e framename são agora usado principalmente com iframes.

Enviando arquivo (Uploading Files)

Imagens e arquivos podem ser carregados / enviados ao servidor, definindo o atributo enctype da tag do formulário como multipart / formdata. enctype especifica como os dados do formulário seriam codificados durante o envio ao servidor.

Exemplo:

```
<form action="envio.js" method="post"  enctype="multipart/form-data">
        <input type="file" name="foto" />
        <input type="submit" value="Enviando" />
</form>
```

Agrupando campos de entradas

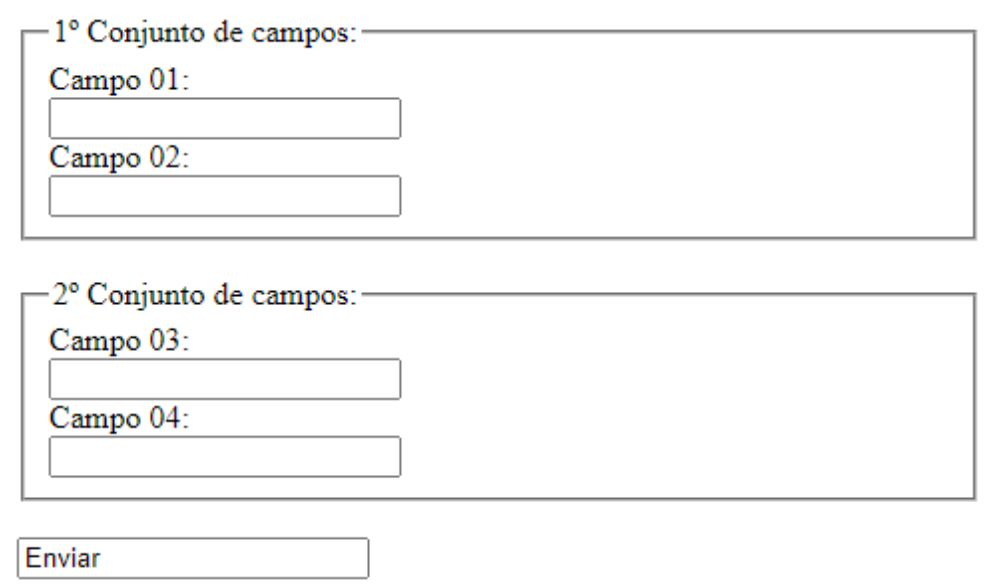
Ao criar um formulário, você pode agrupar alguns campos de entrada em um grupo para ajudar a organizar o layout do formulário. Isso pode ser feito usando a tag Aqui está um exemplo para usá-lo.

Para cada conjunto de campos, você pode definir uma legenda para o conjunto usando a tag LEGEND TEXT

Exemplo:

```
<form>
   <fieldset>
         <legend>1º Conjunto de campos:</legend>
          Campo 01: </br>
          <input type="text"/></br>
          Campo 02:</br>
          <input type="text" />
    </fieldset></br>

     <fieldset>
          <legend>2º Conjunto de campos:</legend>
          Campo 03: </br>
          <input type="text"/></br>
          Campo 04:</br>
          <input type="text" /></br>
      </fieldset></br>
      <input text="submit" value="Enviar"  />
</form>
```

Resultado



Suporte do navegador

As versões mais recentes do Chrome, IE, Edge, FireFox, Safari e Opera também suportam a tag

Capítulo 20: Elemento

O elemento **div** em HTML é um elemento contêiner que encapsula outros elementos e pode ser usado para agrupar e partes separadas de uma página da web. Uma **div** por si só não representa nada inerentemente, mas é uma ferramenta poderosa. Este tópico aborda o objetivo e as aplicações desse elemento **div**.

Uso básico

O elemento <**div**> geralmente não possui significado semântico específico por si só, simplesmente representando uma divisão e é normalmente usado para agrupar e encapsular outros elementos em um documento HTML e separar aqueles de outros grupos de conteúdo. Como tal, cada <**div**> é melhor descrito pelo seu conteúdo.

```
<div>
    <p>Aqui você coloca um parágrafo</p>
</div>
```

O elemento div normalmente é um elemento no nível do bloco, o que significa que separa um bloco de um documento HTML e ocupando a largura máxima da página. Os navegadores normalmente têm a seguinte regra CSS padrão:

```
div {
    display: block;
}
```

É fortemente incentivado pelo The World Wide Web Consortium (W3C) a visualizar o elemento div como um elemento de último recurso, para quando nenhum outro elemento for adequado. O uso de elementos mais apropriados em vez do elemento div, leva a melhor acessibilidade para os leitores e facilidade de manutenção para os autores

Por exemplo, uma postagem no blog seria marcada usando <article>, um capítulo usando <section>, os auxílios de navegação de uma página usando <nav> e um grupo de controles de formulário usando <fieldset>.

Os elementos div podem ser úteis para fins estilísticos ou para agrupar vários parágrafos em uma seção que todos devem ser anotado de maneira semelhante

Aninhamento (Nesting)

É uma prática comum colocar vários dentro de outra . Isso geralmente é chamado de "aninhamento" elementos e permite dividir ainda mais os elementos em subseções ou auxiliar os desenvolvedores com o estilo CSS.

A é usada para agrupar dois elementos ; cada um contendo um elemento <p>

```
<div class="elemento-exterior">
    <div class="elemento-interior">
        <p>Aqui você coloca um parágrafo</p>
    </div>

    <div class="elemento-interior">
        <p>Aqui você coloca um parágrafo</p>
    </div>
</div>
```

Isso produzirá o seguinte resultado (estilos CSS aplicados para maior clareza):

Aninhando de elementos inline e de bloco ao aninhar elementos, lembre-se de que existem inline e elementos de bloco. Enquanto os elementos de bloco "adicionam uma quebra de linha em segundo plano", o que significa que outros elementos aninhados são mostrado na próxima linha automaticamente, os elementos embutidos podem ser posicionados um ao lado do outro por padrão.

**Evite aninhamento profundo **

Um layout profundo e frequentemente usado de contêiner aninhado mostra um estilo de codificação incorreto. Cantos arredondados ou algumas funções semelhantes geralmente criam esse código HTML. Para a maior parte da última geração navegadores existem contrapartes CSS3. Tente usar o mínimo possível de elementos HTML para aumentar o conteúdo a ser marcado e reduza o carregamento da página, resultando em uma melhor classificação nos mecanismos de pesquisa.

div section O elemento não deve ser aninhado a uma profundidade superior a 6 camadas.

Capítulo 21: Elementos de divisão

Elemento <Nav>

O elemento <nav> deve ser usado principalmente para seções que contêm os principais blocos de navegação para o website, isso pode incluir links para outras partes da página da web (por exemplo, âncoras para um sumário) ou outras páginas inteiramente.

Itens embutidos

```
<nav>
   <a href="https://tipscode.com.br"> TipsCode </a>
   <a href="https://youtube.com/tipscode">Canal do TipsCode </a>
   <a href="https://instagram.com/tipscodeoficial"> Instagram do TipsCode </a>
</nav>
```

Use itens da lista quando necessário

Se o conteúdo representar uma lista de itens, use a tag <ul> para mostrar isso e aprimorar a experiência do usuário.

Observe o role = "navigation", mais sobre isso abaixo.

```
<nav role="navigation">
     <ul>
        <li> <a href="https://tipscode.com.br"> TipsCode </a> </li>
       <li> <a href="https://youtube.com/tipscode">Canal do TipsCode </a> </li>
       <li><a href="https://instagram.com/tipscodeoficial"> Instagram do TipsCode </a> </li>
    </ul>
</nav>
```

Evite uso desnecessário

Os elementos <footer> podem ter uma lista de links para outras partes do site (FAQ, T&C, etc.). O elemento rodapé sozinho é suficiente neste caso, você não precisa quebrar ainda mais seus links com um elemento <nav> no <footer>.

```
<!-- o <nav> não é necessário no <footer> -->
<footer>
      <nav>
            <a href="#">...</a>
      </nav>
</footer>

<!-- Somente o rodapé é suficiente -->
<footer>
      <a href="#">...</a>
</footer>
```

Nota:

- descendentes do elemento <main> não são permitidos em um <nav>

A adição de uma função ARIA role = "navigation" ao elemento <nav> é recomendada para ajudar os agentes do usuário que não
suporte HTML5 e também para fornecer mais contexto para aqueles que o fazem.

<nav role = "navigation"> <! - ... -> </nav>

Leitores de tela: (software que permite que usuários cegos ou com deficiência visual naveguem no site)
Os agentes do usuário, como os leitores de tela, interpretarão o elemento <nav> de maneira diferente, dependendo da requisitos.

- Isso poderia dar ao elemento <nav> uma prioridade mais alta ao renderizar a página
- Isso pode atrasar a renderização do elemento
- Poderia adaptar a página de uma maneira específica para atender às necessidades do usuário exemplo: aumente os links de texto dentro dos elementos <nav> para alguém com deficiência visual.

Elemento <Article>

O elemento <article> contém conteúdo independente, como artigos, postagens de blog, comentários de usuários ou um elemento interativo. widget que pode ser distribuído fora do contexto da página, por exemplo, por RSS.

- Quando os elementos do artigo são aninhados, o conteúdo do nó interno do artigo deve estar relacionado ao artigo externo elemento.

Um blog (seção) com várias postagens (artigo) e comentários (artigo) pode se parecer com isso.

```
<section>
      <!-- Cada postagem de blog individual é um <article> -->
      <article>
            <header>
                  <h1>Artigo do Tipscode</h1>
                  <time date="07/07/2020">07/07/2020</time>
            </header>

            <p>O elemento article representa um artigo ou documento independente.</p>
            <p>O elemento de seção representa um agrupamento de conteúdo.</p>
<!-- Somente o rodapé é suficiente -->

             <section>
                  <h2>Comentário <small>Relacionado a esse artigo</small></h2>

                  <!-- Comentário relacionado também é um artigo independente -->
                  <article id="comentario-usuario-1">
                        <p>Muito bom!</p>
                        <footer><p>... </p> <time>....</time></footer>
                   </article>
              </section>
         </article>

</section>

<!--O conteúdo não relacionado ao blog ou postagens deve estar fora da seção. -->
<footer>
      <p>Este conteúdo não deve estar relacionado ao blog.</p>
</footer>
```

Quando o conteúdo principal da página (excluindo cabeçalhos, rodapés, barras de navegação etc.) é simplesmente um grupo de elementos. Você pode omitir o <article> a favor do elemento <main>.

```
<article>
   <p> Isso não faz sentido, este artigo não tem um `contexto 'real.</p>
</article>
```

Em vez disso, substitua o artigo por um elemento <main> para indicar que este é o conteúdo principal desta página

```
<main>
   <p> Sou o conteúdo principal, não preciso pertencer a um artigo. </p>
</main>
```

Se você usar outro elemento, certifique-se de especificar a função ARIA <main> para interpretação e renderização corretas vários dispositivos e navegadores não HTML5.

```
<section role="main">
   <p>Esta seção é o conteúdo principal desta página.</p>
</section>
```

Notas:

- **descendentes de elementos <main> não são permitidos em um <article>**

Elemento <Main>

O elemento <**main**> contém o conteúdo principal da sua página da web. Esse conteúdo é exclusivo da página individual, e não deve aparecer em nenhum outro local do site. Repetir conteúdo como cabeçalhos, rodapés, navegação, logotipos etc. é colocado fora do elemento.

- O elemento <**main**> só deve ser usado no máximo uma vez em uma única página.
- O elemento <**main**> não deve ser incluído como descendente de um artigo, aparte, rodapé, cabeçalho ou navegação elemento.

No exemplo a seguir, estamos exibindo uma única postagem no blog (e informações relacionadas, como referências e comentários).

```
<body>
        <header>
              <h1>Artigo do Tipscode</h1>
         </header>
         <main>
               <h1> Artigo individual</h1>
               <p>Introdução do artigo</p>

               <article>
                     <h1>Referencias<h1>
                     <p>...</p>
               </article>

               <article>
                    <h2>Comentários<h2>
               </article>
          </main>
          <footer>...</footer>
</body>
```

- A postagem do blog está contida no elemento <main> para indicar que este é o conteúdo principal desta página (e portanto, único no site).
- As tags <**header**> e <**footer**> são irmãos do elemento <**main**>.

Notas:

A especificação HTML5 reconhece o elemento <main> como um elemento de agrupamento, e não como um seccionamento elemento.

- Atributos da função ARIA: principal (padrão), apresentação

A adição de um atributo de função ARIA role = "main" a outros elementos destinados a serem usados como conteúdo principal é aconselhado a ajudar agentes de usuários que não suportam HTML5 e também a fornecer mais contexto para aqueles que o fazem.

O elemento <main> por padrão tem a função principal e, portanto, não precisa ser fornecido.

Elemento <Header>

O elemento <header> representa o conteúdo introdutório para seu conteúdo de corte ancestral mais próximo ou corte elemento raiz. Um <cabeçalho> normalmente contém um grupo de auxílios introdutórios ou de navegação.

Nota: O elemento do cabeçalho não está secionando o conteúdo; não apresenta uma nova seção.

Exemplo:

```
<header>
        <h1>Artigo do Tipscode</h1>
 </header>
```

Neste exemplo, o <article> possui um <header>.

```
<article>
      <header>
            <h1>Artigo do Tipscode</h1>
      </header>
      <p>.....</p>
</article>
```

Elemento <Footer>

O elemento <footer> contém a parte do rodapé da página.

Aqui está um exemplo para o elemento <footer> que contém a tag de parágrafo p.

```
<footer>
        <p>Todos os deireitos reservados ao Tipscode</h1>
</header>
```

Elemento <Section>

O elemento <section> representa uma seção genérica para agrupar tematicamente o conteúdo. Cada seção, normalmente, deve poder ser identificado com um elemento de cabeçalho como filho da seção.

- Você pode usar o elemento <section> dentro de um <article> e vice-versa.
- Cada seção deve ter um tema (um elemento de cabeçalho que identifique esta região)
- Não use o elemento <section> como um 'contêiner' de estilo geral.
  Se você precisar de um contêiner para aplicar estilo, use um .

No exemplo a seguir, exibimos uma única postagem no blog com vários capítulos, cada capítulo é uma seção (um conjunto do conteúdo agrupado tematicamente, que pode ser identificado pelos elementos de cabeçalho em cada seção).

```
<article>
      <header>
            <h1>Artigo do Tipscode</h1>
      </header>
      <p>.....</p>
      <section>
            <h3>Cap 01</h3>
            <p>.....</p>
       </section>
       <section>
            <h3>Cap 02</h3>
            <p>.....</p>
        </section>
        <section>
             <h3>Comentários..</h3>
        </section>
</article>
```

Capítulo 22: Navegação <Nav>

Básico da Navegação

As barras de navegação são essencialmente uma lista de links; portanto, os elementos <**ul**> e <**li**> são usados para envolver os links de navegação.

```
<ul>
    <li> <a href="#"> Página Inicial </a> </li>
    <li> <a href="#"> Sobre </a> </li>
    <li> <a href="#"> Contato </a> </li>
</ul>
```

Navegação <nav>

Para criar uma barra de navegação usando o elemento de navegação HTML5, coloque os links na tag de navegação.

```
<nav>
    <li> <a href="#"> Página Inicial </a> </li>
    <li> <a href="#"> Sobre </a> </li>
    <li> <a href="#"> Contato </a> </li>
</nav>
```

Capítulo 23: Elemento <Label>

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
for	Referência ao elemento de ID de destino. Ou seja: para = "sobrenome"
form	HTML5, [obsoleto] Referência ao formulário que contém o elemento de destino. Elementos do rótulo são esperados dentro de um elemento <form>. Se o formulário = "someFormId" for fornecido, isso permitirá que você coloque o rótulo em qualquer lugar do documento.

Sobre o elemento <label>

O elemento <**label**> é usado para referenciar um elemento de ação do formulário.
No escopo da interface do usuário, é usado para facilitar o destino / seleção de elementos como o tipo de rádio ou a caixa de seleção.

<label> como wrapper (Um envolucro)

Pode incluir o elemento de ação desejado

```
<label>
    <input type="checkbox" name="Cidade" />
     Minha cidade
</label>
```

(Ao clicar no texto, a entrada de destino alternará seu estado / valor)

<label> como referência

Usando o atributo for, você não precisa colocar o elemento de controle como descendente do <label>, mas o valor for deve coincidir com o ID

```
<input id="cidade" type="checkbox" name="cidade" />
<label for="cidade" >Minha cidade</label>
```

Nota:

Não use mais de um elemento de controle em um elemento <label>

Uso básico

Formulário simples com etiquetas ...

```
<form action="/logion" method="POIST">

      <label for="username"></label>
      <input id="username" type="text" name="username" />

      <label for="username"></label>
      <input id="username" type="text" name="username" />

      <input id="submit" name="username" />
</form>
// Versão >= 5
<form id="my-form" action="/login" method="POST">

       <input id="username" type="text" name="username" />

      <label for="username"></label>
      <input id="username" type="text" name="username" />
</form>

<label for="username" form="my-form"></label>
```

Capítulo 24: Elemento <Output>

www.tipscode.com.br | Alisson Suassuna	
Atributos	**Descrições**
Global	Atributos disponíveis para qualquer elemento HTML5. Para uma documentação abrangente desses atributos, consulte: Atributos Globais do MDN
name	Uma sequência que representa o nome de uma saída. Como elemento de formulário, a saída pode ser referenciada por seu nome usando a propriedade document.forms. Este atributo também é usado para coletar valores em um envio de formulário.
for	Uma lista separada por espaços de IDs de elementos de formulário (por exemplo, <inputs id = "inp1"> para o valor é "inp1") que o saída destina-se a exibir cálculos para.
form	Uma sequência que representa o <form> associado à saída. Se a saída estiver realmente fora do <form>, este atributo garantirá que a saída ainda pertença ao <form> e esteja sujeita a coleções e envia o referido <form>.

Elemento de saída usando atributos For e Form

A demonstração a seguir apresenta o uso de um elemento <**output**> dos atributos [for] e [form]. Lembre-se <**output**> precisa de JavaScript para funcionar. JavaScript embutido é comumente usado em formulários, como este exemplo demonstra. Embora os elementos <**input**> sejam **type** = "number", seus valores não são números, eles são texto. Então, se você precisar os valores a serem calculados, você deve **converter** cada valor em um número usando métodos como: **parseInt** (), **parseFloat** (), **Number** () etc.

```
<!-- O form1 coletará os valores de in1 e in2 no evento 'input' -->
<!-- O valor out1 será a soma dos valores in1 e in2. -->
<form id="form1" name="form1" oninput="out1.value = parseInt(in1.value, 10) +
parseInt(in2.value, 10)">
    <fieldset>
        <legend> Exemplo de Saída!! </legend>
        <input type="number" id="in1" name="in1" value="0" />
        </br>
        +
        <input type="number" id="in2" name="in2" value="0" />
    </fieldset>
</form>
// Versão >= 5
<form id="my-form" action="/login" method="POST">

      <input id="username" type="text" name="username" />

      <label for="username"></label>
      <input id="username" type="text" name="username" />
</form>

<output name="out1" form="form1" for="inp1 inp2"></output>
```

Capítulo 25: Elemento <Void>

Nem todas as tags HTML têm a mesma estrutura. Embora a maioria dos elementos exija uma tag de abertura, uma tag de fechamento e conteúdo, alguns elementos - conhecidos como elementos **nulos** - exigem apenas uma tag de abertura, pois eles próprios não contêm quaisquer elementos. Este tópico explica e demonstra o uso adequado de elementos nulos em HTML

Elemento fazio (void)

O HTML 4.01 / XHTML 1.0 Strict inclui os seguintes elementos nulos:

- **area** - área definida e clicável em uma imagem
- **base** - especifica um URL base a partir do qual todos os links baseiam
- **br** - quebra de linha
- **col** - coluna em uma tabela [obsoleta]
- **hr** - regra horizontal (linha)
- **img** - imagem
- **input** - campo em que os usuários inserem dados
- **link** - vincula um recurso externo ao documento
- **meta** - fornece informações sobre o documento
- **param** - define parâmetros para plugins

```
<div>
    <a href="https://tipscode.com.br">
        <h3> Compartilhando Conhecimento </h3>
    </a>
    <button onclick="alert=('Olá!')">Olá, Mundo</button>
    <p>Guia completo de HTML</p>
    <ol>
        <li>JavaScript</li>
        <li>CSS</li>
        <li>Nodejs</li>
    </ol>
</div>
```

Observe como cada elemento tem uma tag de abertura, uma tag de fechamento e texto ou outros elementos dentro da abertura e tags de fechamento. As tags nulas, no entanto, são mostradas no exemplo abaixo:

```
<img src="https://tipscode.com.br/tipscode.png" />
</br>
</hr>
<input type="number" placeholder="Qual seu número?" />
```

Com exceção da tag img, todos esses elementos nulos têm apenas uma tag de abertura. A tag img, diferente de qualquer outra tag, possui um fechamento automático / antes do sinal maior que a marca de abertura.

Capítulo 26: Elemento <Midia>

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
width	Define a largura do elemento em pixels.
height	Define a altura do elemento em pixels
<source>	Define os recursos dos arquivos de áudio ou vídeo
track	Define a faixa de texto para elementos de mídia
controls	Exibe controles
autoplay	Iniciar automaticamente a reprodução da mídia
loop	Reproduz a mídia em um ciclo repetido
muted	Reproduz a mídia sem som
posted	Atribui uma imagem a ser exibida até que um vídeo seja carregado

Áudio (Audio)

O HTML5 fornece um novo padrão para incorporar um arquivo de áudio em uma página da web. Você pode incorporar um arquivo de áudio a uma página usando o elemento <audio>:

```
<audio controls>
    <source src="arquivo.mp3" type="audio/mpeg" />
</audio/>
```

Vídeo (video)

Você também pode incorporar um vídeo a uma página da Web usando o elemento <video>:

```
<video width="500" height="700" controls>
    <source src="video.mp4" type="video/mp4" />
</video/>
```

Usando o elemento `<video>` e `<audio>` para exibir conteúdo de áudio / vídeo

Use o elemento HTML <**video**> para incorporar o conteúdo de vídeo / áudio em um documento. O elemento de áudio/ áudio contém uma ou mais fontes de vídeo / áudio. Para especificar uma fonte, use o atributo src ou o elemento <source>; a O navegador escolherá o mais adequado.

Exemplo de tag de áudio:

```
<!-- Exemplo simples -->
<video src="arquivovideo.mp4" autoplay poster="bannervideo.jpg">
    Desculpe, seu navegador não suporta vídeos incorporados,
    mas não se preocupe, você pode <a href="arquivovideo.mp4">Faça o download aqui</a>
</video/>

<!-- Vídeo com legendas -->
<video src="tipscode.mp4">
    <track kind="subtitles" src="tipscode.pt.vtt" srclang="pt-br" label="Portugues" />
    <track kind="subtitles" src="tipscode.en.vtt" srclang="en" label="English" />
</video>
<!-- Exemplo de vídeo simples -->
<video width="400" controls poster="https://tipscode.org/download/mp4">
    <source src="https://tipscode.com.br/download/tips.mp4" type="video/mp4" />
    <source src="https://tipscode.com.br/download/tips.webm" type="video/webm" />
    <source src="https://tipscode.com.br/download/tips.ogg" type="video/ogg" />
    Seu navegador não suporta a tag de vídeo HTML5.
</video>
```

Exemplo de tag de áudio:

```
<!-- Exemplo simples de audio -->
<audio src="https://tipscode.com.br/musica.mp3" autoplay>
    Seu navegador não suporta o elemento de <code>audio</code>
</audio>

<!-- Reprodução de áudio com legendas  -->
<audio src="musica.ogg">
        <track kind="captions" src="musica.en.vtt" srclang="en" label="English" />
        <track kind="captions" src="musica.pt.vtt" srclang="pt-br" label="Portugues" />
</audio>
```

Cabeçalho ou plano de fundo usando a tag <video>

Adicionando um vídeo que será reproduzido automaticamente em um loop e não possui controles ou som. Perfeito para um cabeçalho ou plano de fundo de vídeo.

```
<video width="1280"  height="720" autoplay muted loop poster="video.jpg" id="videobg">
    <source src="tips.mp4" type="video/mp4" />
    <source src="tips.webm" type="video/webm" />
    <source src="tips.ogg" type="video/ogg" />
</video>
```

Este CSS fornece um substituto se o vídeo não puder ser carregado. Observe que é recomendável usar o primeiro quadro do video como o poster video.jpg.

```
#.videobg {
  background: url(video.jpg) no-repeat;
  background: cover;
}
```

Capítulo 27: Elemento <Progress>

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
max	Quanto trabalho a tarefa requer no total
value	Quanto do trabalho já foi realizado
position	Este atributo retorna a posição atual do elemento <progress>
labels	Este atributo retorna uma lista de rótulos de elemento <progress> (se houver)

Progress

O elemento <**progress**> é novo no HTML5 e é usado para representar o progresso de uma tarefa

```
<progress value="1280"  max="100"></progress>
```

Isso cria uma barra cheia de 22%

Alterando a cor de uma barra de progresso

As barras de progresso podem ser estilizadas com o seletor de progresso [valor].
Este exemplo fornece uma barra de progresso com uma largura de 250 px e uma altura de 20 px

```
progress[value] {
  width: 250px;
  height: 20px;
}
```

As barras de progresso podem ser especialmente difíceis de estilizar.

Chrome / Safari / Opera

Esses navegadores usam o seletor -webkit-aparecimento para estilizar a marca de progresso.
Para substituir isso, podemos redefinir o aparência.

```
progress[value] {
  -webkit-appearance: none;
  appearance: none;
}
```

Agora, podemos estilizar o próprio contêiner

```
progress[value]:: -webkit-progress-bar{
  background-color: red;
}
```

Firefox

O Firefox estiliza a barra de progresso de maneira um pouco diferente. Temos que usar esses estilos

```
progress[value] {
   -moz-appearance: none;
   appearance: none;
   border: none;
}
```

Internet Explorer

O Internet Explorer 10+ suporta o elemento de progresso. No entanto, ele não suporta a cor de fundo
propriedade. Você precisará usar a propriedade color.

```
progress[value] {
   -webkit-appearance: none
   -moz-appearance: none;
   appearance: none;
   border: none;

   width: 250px;
   height: 20px;

   color: red;
}
```

HTML Fallback

Para navegadores que não suportam o elemento progress, você pode usar isso como uma solução alternativa.

```
<progress value="20"  max="100">
        <div class="progress-bar">
                <span style="20%;">Progresso</span>
        </div>
</progress>
```

Os navegadores que suportam a tag de progresso ignoram a div aninhada dentro. Navegadores herdados que não conseguem identificar o A tag de progresso renderizará a div.

Capítulo 28: Elemento <Selection>

Select Menu

O elemento <**select**> gera um menu suspenso no qual o usuário pode escolher uma opção.

```
<select name=" ">
        <option value="01"> Primeiro </option>
        <option value="02"> Segundo </option>
        <option value="03"> Terceiro </option>
        <option value="04"> Quarto </option>
</select>
```

Alterando o tamanho

Você pode alterar o tamanho do menu de seleção com o atributo **size**. Um tamanho de 0 ou 1 exibe o menu de estilo suspenso padrão. Um tamanho maior que 1 converterá o menu suspenso em uma caixa que exibe muitas linhas, com uma opção por linha e uma barra de rolagem para rolar pelas opções disponíveis.

```
<select name=" " size="04"></select>
```

Menus de seleção com várias opções

Por padrão, os usuários podem selecionar apenas uma única opção. A adição do atributo **múltiplo** permite que os usuários selecionem vários opções de uma só vez e envie todas as opções selecionadas com o formulário. O uso do atributo múltiplo converte automaticamente o menu suspenso em uma caixa como se tivesse um tamanho definido. O tamanho padrão quando isso ocorre é determinado pelo navegador específico que você está usando e não é possível alterá-lo novamente para um menu de estilo suspenso enquanto permite várias seleções.

```
<select name=" " multiple></select>
```

Ao usar o atributo múltiplo, há uma diferença entre usar 0 e 1 para o tamanho, enquanto nenhuma diferença existe quando não estiver usando o atributo.

Usar 0 fará com que o navegador se comporte da maneira padrão que foi programado para fazer. Usar 1 definirá explicitamente o tamanho da caixa resultante para apenas uma linha de altura.

Options

As opções dentro de um menu de seleção são as que o usuário selecionará. A sintaxe normal de uma opção é a seguinte: segue:

```
<option> Algumas opções... </option>
```

No entanto, é importante observar que o texto dentro do próprio elemento <**option**> nem sempre é usado e, essencialmente, torna-se o valor padrão para atributos que não são especificados.

Os atributos que controlam a aparência e a função reais da opção são value e label. O label representa o texto que será exibido no menu suspenso (o que você está vendo e clicará para selecionar isto). O value representa o texto que será enviado junto com o envio do formulário. Se um desses valores for omitido, em vez disso, usa o texto dentro do elemento como valor. Portanto, o exemplo que demos acima pode ser "expandido" para isso:

```
<option label="algumas opções" value="Algumas Opções"> </option>
```

Observe a omissão do texto interno e da tag final, que não são necessárias para realmente construir uma opção dentro do menu. Se eles fossem incluídos, o texto interno seria ignorado porque os dois atributos já estão especificados e o texto não é necessário. No entanto, você provavelmente não verá muitas pessoas escrevendo-as dessa maneira. A maneira mais comum está escrito com um valor que será enviado ao servidor, juntamente com o texto interno que eventualmente se torna o atributo, assim:

```
<option value="option1"> Algumas Opções.. </option>
```

Selecionando uma opção por padrão Você também pode especificar uma determinada opção a ser selecionada no menu por padrão, anexando o atributo selecionado a ele. Por padrão, se nenhuma opção for especificada como selecionada no menu, a primeira opção no menu será selecionada. Se mais de uma opção tiver o atributo selecionado anexado, a última opção presente no menu com o atributo será o selecionado por padrão.

```
<option value="option1" selected> Algumas Opções.. </option>
```

Se você estiver usando o atributo em um menu de seleção com várias opções, todas as opções com o atributo serão selecionadas por padrão, e nenhuma será selecionada se nenhuma opção tiver o atributo

```
<select multiple>
        <option value="option01"> Primeiro </option>
        <option value="option02"> Segundo </option>
</select>
```

Grupo de Opções (Options Groups)

Você pode agrupar suas opções ordenadamente em um menu de seleção para fornecer um layout mais estruturado em um longo lista de opções usando o elemento <optgroup>.

A sintaxe é muito básica, basta usar o elemento com um atributo label para identificar o título do grupo e
contendo zero ou mais opções que devem estar dentro desse grupo.

```
<select name=" ">
        <option value="Desenvolvimento Web"> Desenvolvimento Web </option>
        <optgroup label="Linguagens"> Linguagens </option>
            <option value="JavaScript"> JavaScript </option>
            <option value="Java"> Java </option>
        </optgroup>

        <optgroup label="Framework / Lib">
             <option value="React"> React </option>
             <option value="Angular"> Angular </option>
        </optgroup>
</select>
```

Ao usar grupos de opções, nem todas as opções precisam estar contidas em um grupo. Desabilitando também um grupo de opções desativará todas as opções dentro do grupo e não é possível reativar manualmente uma única opção dentro de um grupo desativado.

Datalist

A tag <**datalist**> especifica uma lista de opções predefinidas para um elemento <**input**>. Ele fornece um "preenchimento automático" recurso nos elementos <**input**>. Os usuários verão uma lista suspensa de opções enquanto escrevem.

```
<input list="Linguagens">
        <datalist id="linguagens">
             <option value="JavaScript"> </option>
             <option value="JavaScript"> </option>
             <option value="Java"> </option>
             <option value="Python"> </option>
             <option value="C++"> </option>
        </datalist>
```

Capítulo 29: Elemento <Embed>

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
src	Endereço do recurso
type	Tipo de recurso incorporado
width	Dimensão horizontal
height	Dimensão vertical

Uso básico

A tag incorporar é nova no HTML5. Este elemento fornece um ponto de integração para um externo (normalmente não HTML) aplicativo ou conteúdo interativo.

```
<embed src="ohoh.swf" />
```

Definindo o Tipo MINE

O tipo MIME deve ser definido usando o atributo type

```
<embed src="video/mp4" src="video.mp4" width="650" height="410" />
```

Capítulo 30: Elemento <IFrames>

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
name	Define o nome do elemento, a ser usado com uma tag a para alterar o src do iframe.
width	Define a largura do elemento em pixels.
height	Define a altura do elemento em pixels.
src	Especifica a página que será exibida no quadro
srcdoc	Especifica o conteúdo que será exibido no quadro, assumindo que o navegador seja compatível. o o conteúdo deve ser HTML válido.
sandbox	Quando definido, o conteúdo do iframe é tratado como sendo de origem e recursos exclusivos incluindo scripts, plugins, formulários e pop-ups serão desativados. Restrições podem ser seletivamente relaxado adicionando uma lista de valores separados por espaço. Consulte a tabela em Comentários para obter os valores possíveis.

allowfullscreen Se permite que o conteúdo do **iframe** use **requestFullscreen** ()

Noções básicas do elemento IFrame

O termo "IFrame" significa quadro embutido. Pode ser usado para incluir outra página na sua página. Isso produzirá um pequeno quadro que mostra o conteúdo exato do arquivo base.html.

```
<iframe src="base.html"></iframe>
```

Sandboxing

A seguir, incorpora uma página da web não confiável com todas as restrições ativadas

```
<iframe  sandbox src="https://tipscode.com.br"></iframe>
```

Para permitir que a página execute scripts e envie formulários, inclua allow-scripts e allow-forms no atributo sandbox.

```
<iframe  sandbox="allow-scripts allow-forms" src="https://tipscode.com.br"></iframe>
```

Se houver conteúdo não confiável (como comentários do usuário) no mesmo domínio da página da Web pai, um iframe poderá ser usado para desativar scripts enquanto ainda permite que o documento pai interaja com seu conteúdo usando JavaScript.

```
<iframe  sandbox="allow-same-origin allow-top-navigation"
src="https://tipscode.com.br/tips/page1"></iframe>
```

O documento pai pode adicionar ouvintes de eventos e redimensionar o IFrame para ajustar seu conteúdo. Isso, junto com a navegação de permissão superior, pode fazer com que o iframe na área de areia pareça fazer parte do documento pai.

Essa sandbox não substitui a entrada de higienização, mas pode ser usada como parte de uma estratégia de defesa em profundidade.

Lembre-se também de que este sandbox pode ser subvertido por um invasor convencendo um usuário a visitar a fonte do iframe diretamente. O cabeçalho HTTP da Política de Segurança de Conteúdo pode ser usado para atenuar esse ataque.

Tamanhos

O IFrame pode ser redimensionado usando os atributos width e height, onde os valores são representados em pixels (HTML 4.01 permitiram valores percentuais, mas o HTML 5 permite apenas valores em pixels CSS).

```
<iframe src="base.html" width="800" height="600"></iframe>
```

Usando o atributo srcdoc

O atributo srcdoc pode ser usado (em vez do atributo src) para especificar o conteúdo exato do iframe como um documento HTML inteiro. Isso produzirá um IFrame com o texto "IFrames are cool!"

```
<iframe srcdoc="<p>IFrame </p>" ></iframe>
```

Se o atributo srcdoc não for suportado pelo navegador, o IFrame voltará a usar o atributo src, mas se os atributos src e srcdoc estiverem presentes e forem suportados pelo navegador, o srcdoc terá precedência.

```
<iframe srcdoc="<p>IFrame </p>" src="base.html" ></iframe>
```

No exemplo acima, se o navegador não suportar o atributo srcdoc, ele exibirá o conteúdo de a página base.html.

Usando âncoras com IFrames

Normalmente, uma mudança de página da Web em um Iframe é iniciada a partir do Iframe, por exemplo, clicando em um link dentro o Ifame. No entanto, é possível alterar o conteúdo de um IFrame de fora do IFrame. Você pode usar uma âncora tag cujo atributo href está definido para o URL desejado e cujo atributo de destino está definido para o atributo de nome do iframe

```
<iframe  src="homepase.html" name="meuIframe"></iframe>
<a href="www.tipscode.com.br" target="meuIframe">Visite clicando aqui!</a>
```

Capítulo 31: Idioma do Conteúdo

Base do idioma do conteúdo

É uma boa prática declarar o idioma principal do documento no elemento html:

```
<html lang="pt-br">
```

Se nenhum outro atributo lang for especificado no documento, significa que tudo (ou seja, conteúdo e atributo do elemento valores de texto) está nesse idioma.

Se o documento contiver partes em outros idiomas, essas partes deverão obter seus próprios atributos lang para "substituir" a declaração de idioma.

Elemento de idioma

O atributo lang é usado para especificar o idioma do conteúdo do elemento e os valores do texto do atributo:

```
<p lang="pt-br"> Esse conteúdo está em Português</p>
<p lang="en"> Esse conteúdo está em Inglês</p>
```

A declaração de idioma é herdada:

```
<div lang="pt-br">
    <p lang="en"> Esse conteúdo está em Inglês</p>
    <p title="Título desse conteúdo"> blablabla </p>
</div>
```

Elemento com vários idiomas

Você pode "substituir" uma declaração de idioma:

```
<p lang="pt-br">Esse texto está em pt-br <span lang="en"> Hello</span></p>
```

Capítulo 32: SVG

SVG significa Scalable Vector Graphics. SVG é usado para definir gráficos para a Web

O elemento HTML <**svg**> é um contêiner para gráficos SVG.

O SVG possui vários métodos para desenhar caminhos, caixas, círculos, texto e imagens gráficas.

SVG interno

O SVG pode ser gravado diretamente em um documento HTML. O SVG embutido pode ser estilizado e manipulado usando CSS e JavaScript.

```
<body>
        <svg class="attention" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 0 1000 1000" >
            <path id="attention"
d="m571,767l0,-106q0,-8,-5,-13t-12,-5l-108,0q-7,0,-12,5t-
5,13l0,106q0,8,5,13t12,6l108,0q7,0,12,-6t5
,-13Zm-1,-208l10,-257q0,-6,-5,-10q-7,-6,-14,-6l-122,0q-7,0,-14,6q-
5,4,-5,12l9,255q0,5,6,9t13,3l103,
0q8,0,13,-3t6,-9Zm-7,-522l428,786q20,35,-1,70q-10,17,-26,26t-35,10l-858,0q-18,0,-35,-10t-
26,-26q-21
,-35,-1,-70l429,-786q9,-17,26,-27t36,-10t36,10t27,27Z" />
        </svg>
</body>
```

O SVG embutido acima pode ser estilizado usando a classe CSS correspondente:

```
.attentio {
    fill: red;
    width: 50px;
    height: 50px;
}
```

O resultado fica assim:

Incorporando arquivos SVG externos em HTML

Você pode usar os elementos <img> ou <object> para incorporar elementos SVG externos. Definir a altura e largura é opcional, mas é altamente recomendado.

Usando o elemento de imagem

```
<img src="attention.svg" width="50" height="50" />
```

O uso de <img> não permite estilizar o SVG usando CSS ou manipulá-lo usando JavaScript.

Usando o elemento de objeto

```
<object type="image/svg+xml" data="attention.svg" width="50" height="50" />
```

Ao contrário de <**img**>, <object> importa diretamente o SVG para o documento e, portanto, pode ser manipulado usando Javascript e CSS.

Incorporando SVG usando CSS

Você pode adicionar arquivos SVG externos usando a propriedade background-image, como faria com qualquer outra imagem.

HTML:

```
<div class"attention.svg"></div>
```

CSS:

```
.attentio {
    background-image: url(attentio.svg);
    background-size: 100% 100%;
    height: 50px;
    width: 50px;
}
```

Você também pode incorporar a imagem diretamente em um arquivo css usando um URL de dados:

```
background-image:
url(data:image/svg+xml,%3Csvg%20xmlns%3D%22http%3A%2F%2Fwww.w3.org%2F2000
%2Fsvg%22%20xmlns%3Axlink
%3D%22http%3A%2F%2Fwww.w3.org%2F1999%2Fxlink%22%20viewBox%3D%220%200
%201000%201000%22%20%3E%0D%0A%
3Cpath%20id%3D%22attention%22%20d%3D%22m571%2C767l0%2C-106q0%2C-8%2C-
5%2C-13t-12%2C-5l-108%2C0q-7%
2C0%2C-12%2C5t-
5%2C13l0%2C106q0%2C8%2C5%2C13t12%2C6l108%2C0q7%2C0%2C12%2C-6t5%2C-
13Zm-1C-1%2C70q-10%2C17%2
C-26%2C26t-35%2C10l-858%2C0q-18%2C0%2C-35%2C-10t-26%2C-26q-21%2C-35%2C-
1%2C-70l429%2C-786q9%2C-17%
2C26%2C-27t36%2C-10t36%2C10t27%2C27Z%22%20%2F%3E%0D%0A%3C%2Fsvg%3E)
```

Capítulo 33: Canvas

Atributo	Descrição
height	Especifica a altura da tela
width	Especifica a largura da tela

Exemplos básicos

O elemento canvas foi introduzido no HTML5 para desenhar gráficos

```
<canvas id="meuCanvas">
        Não é possível exibir gráficos. O Canvas não é suportado pelo seu navegador (IE <9)
</canvas>
```

O exemplo acima criará um elemento <**canvas**> HTML transparente de 300 × 150 px de tamanho.

Você pode usar o elemento canvas para desenhar coisas incríveis, como formas, gráficos, manipular imagens, criar imagens atraentes jogos etc. com JavaScript.

O objeto da superfície da camada desenhável 2D da tela é conhecido como CanvasRenderingContext2D; ou de um HTMLCanvasElement usando o método .getContext ("2d"):

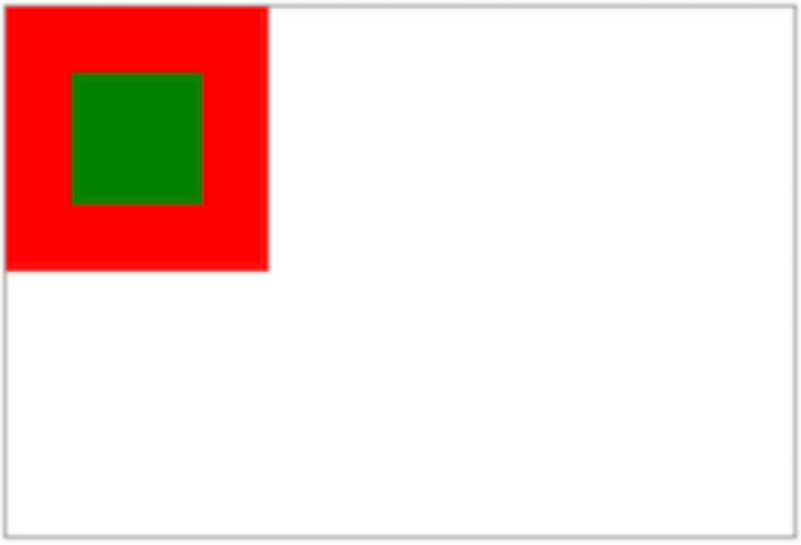
```
let ctx = document.querySelector("meuCanvas").getContext("2d")
 // agora podemos nos referir ao contexto da camada 2D da tela usando `ctx`

ctx.fileStyle = "F00";
ctx.fileRect(0, 0, ctx.canvas.width, ctx.canvas.height ) // x, y, width, height

ctx.fillStyle = "#000"
ctx.fillText("Minha tela vermelha com algum texto em preto", 24, 32); // text, x, y
```

Desenhando dois retângulos em <canvas>

```
<html lang="pt-br">
<head>
        <meta charset="utf-8" />
        <title> Desenhando 2 retângulos </title>
        <style>
                canvas {
                        border: 1px solid gray;
                }
        </style>
        <script async>
                window.onload = init; // chame init () quando a janela estiver completamente carregada
</head>
```

```
        function init() {
            // 01 obter referência ao elemento <canvas>
            let canvas = document.querySelector("canvas")

            // 02 obter referência ao contexto de desenho e à API de desenho
            let ctx = canvas.getContext("2d")

            // 03 todas as operações de preenchimento estão agora em vermelho
            ctx.fillStyle = "red"

            // 04 preencha um retângulo de 100x100 em x = 0, y = 0
            ctx.fillRect(0, 0, 100, 100)

            // 05 todas as operações de preenchimento estão agora em verde
            ctx.fillStyle = "green"

            // 06 preencha um retângulo de 50x50 em x = 25, y = 25
            ctx.fillRect(25, 25, 50, 50)
        }
    </script>
</head>
<body>
    <canvas widtth=300 height=200> Seu navegador não suporta tela. </canvas>
</body>
</html>
```

Veja o Resultado:



Capítulo 34: Informações <Meta>

As metatags nos documentos HTML fornecem informações úteis sobre o documento, incluindo descrição, palavras-chave, autor, datas das modificações e cerca de 90 outros campos. Este tópico aborda o uso e a finalidade dessas tags.

Informações da Página

Nome da Aplicação

Fornecendo o nome do aplicativo Web que a página representa.

```
<meta name="nome-aplicacao" content="MapaRua" />
```

Se não for um aplicativo da Web, a metatag do nome do aplicativo não deve ser usada.

autor

Defina o autor da página:

```
<meta name="author" content="Alisson Suassuna" />
```

Apenas um nome pode ser dado.

descrição

Defina a descrição da página:

```
<meta name="description" content="Descrição da Página" />
```

A meta tag de descrição pode ser usada por vários mecanismos de pesquisa enquanto indexa sua página da web para pesquisa objetivo. Geralmente, a descrição contida na metatag é o breve resumo que aparece sob o
título principal da página / site nos resultados do mecanismo de pesquisa. O Google geralmente usa apenas as primeiras 20 a 25 palavras de seu descrição.

generator

```
<meta name="generator" content="Gerador HTML " />
```

Identifica um dos pacotes de software usados para gerar o documento. Apenas para ser usado em páginas em que a marcação é gerada automaticamente.

palavras-chave

Defina palavras-chave para mecanismos de pesquisa (separados por vírgula):

```
<meta name="keywords" content="Palavra1, Palavra2" />
```

Codificação de Caracteres

O atributo charset especifica a codificação de caracteres para o documento HTML e precisa ser uma codificação de caracteres válida (exemplos incluem windows-1252, ISO-8859-2, Shift_JIS e UTF-8). UTF-8 (Unicode) é o mais amplamente usado e deve ser usado para qualquer novo projeto.

Versão = 5

```
<meta charset="UTF-8" />
<meta charset="ISO-8859-1" />
```

Todos os navegadores sempre reconheceram o formulário <meta charset>, mas se, por algum motivo, você precisar que sua página seja HTML 4.01 válido, você pode usar o seguinte:

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<meta http-equiv="content-type" content="text/html, charset=ISO-8859-1" />
```

Consulte também o Padrão de codificação, para visualizar todos os rótulos de codificação de caracteres disponíveis que os navegadores reconhecem

Robots

O atributo robots, suportado por vários principais mecanismos de pesquisa, controla se as aranhas dos mecanismos de pesquisa são permitido indexar uma página ou não e se eles devem seguir os links de uma página ou não.

```
<meta name="robots" content="noindex" />
```

Este exemplo instrui todos os mecanismos de pesquisa a não mostrar a página nos resultados da pesquisa. Outros valores permitidos são:

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
all	Padrão. Equivalente ao índice, siga. Ver nota abaixo.
noindex	Não indexe a página
nofollow	Não siga os links nesta página
follow	Os links na página podem ser seguidos. Ver nota abaixo.
none	Equivalente a noindex, nofollow.
noarchive	Não disponibilize uma versão em cache desta página nos resultados da pesquisa.
nocache	Sinônimo de noarchive usado por alguns bots, como o Bing.
nosnippet	Não mostre um trecho desta página nos resultados da pesquisa.
noodp	Não use os metadados desta página do projeto Open Directory para títulos ou trechos nos resultados da pesquisa.
notranslate	Não ofereça traduções desta página nos resultados da pesquisa.
noimageindex	Não indexe imagens nesta página.
unavailable_after [RFC-850date/time]	Não mostre esta página nos resultados da pesquisa após a data / hora especificada. o a data / hora deve ser especificada no formato RFC 850.

Nota: Definir explicitamente o índice e / ou seguir, embora valores válidos, não é necessário, pois praticamente todos os mecanismos de pesquisa assumirá que eles podem fazê-lo, se não explicitamente impedidos de fazê-lo. Semelhante a como o arquivo robots.txtopera, os mecanismos de pesquisa geralmente procuram apenas coisas que eles não têm permissão para fazer. Apenas declarando coisas uma pesquisa O mecanismo não pode fazer isso também evita a indicação acidental de opostos (como índice, ..., noindex) que nem todos

Social Media

Open Graph é um padrão para metadados que estende as informações normais contidas no cabeçalho de um site marcação. Isso permite que sites como o Facebook exibam informações mais profundas e ricas sobre um site em um formato estruturado. Essas informações são exibidas automaticamente quando os usuários compartilham links para sites que contêm

Metadados OG no Facebook.

Facebook / Gráfico Aberto

```
<meta property="fb:app_id" content="123456789" />
<meta property="og:url" content="https://example.com/page.html" />
<meta property="og:type" content="website" />
<meta property="og:title" content="Content Title" />
<meta property="og:image" content="https://example.com/image.jpg" />
<meta property="og:description" content="Description Here" />
<meta property="og:site_name" content="Site Name" />
<meta property="og:locale" content="en_US" />
<meta property="article:author" content="" />
<!-- Facebook: https://developers.facebook.com/docs/sharing/webmasters#markup -->
<!-- Open Graph: http://ogp.me/ -->
```

- Guia de compartilhamento para WebMasters;
- Open Graph Protocol.

Facebook / Artigos Instantâneos

```
<meta charset="utf-8" />
<meta property="op:markup_version" content="v1.0" />

<!-- O URL da versão web do seu artigo -->
<link rel="canonical" href=https://tipscode.com.br/artigo.html" />

<!-- O estilo a ser usado para este artigo -->
<meta property="fb:article.style" content="meu artigo" />
```

- Facebook / artigo instantâneos: criado artigos.
- Artigo Instanâneos: formatações

O Twitter usa sua própria marcação para metadados. Esses metadados são usados como informações para controlar como os tweets são exibido quando eles contêm um link para o site.

Twitter

```
<meta name="twitter:card" content="summary" />
<meta name="twitter:site" content="@site_account" />
<meta name="twitter:creator" content="@individual_account" />
<meta name="twitter:url" content="https://example.com/page.html" />
<meta name="twitter:title" content="Content Title" />
<meta name="twitter:description" content="Content description less than 200 characters" />
<meta name="twitter:image" content="https://example.com/image.jpg" />
```

- Otimize os twitter;

Responsividade de Layout Mobile

Sites comuns otimizados para dispositivos móveis usam a tag <meta name = "**viewport**"> como esta:

```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

O elemento viewport fornece ao navegador instruções sobre como controlar as dimensões e o dimensionamento da página com base no dispositivo que você está usando.

No exemplo acima, content = "width = width do dispositivo significa que o navegador renderizará a largura da página em a largura de sua própria tela. Portanto, se essa tela tiver 480px de largura, a janela do navegador terá 480px de largura. initialscale = 1 mostra que o zoom inicial (que é 1 neste caso, significa que não aumenta o zoom).

Abaixo estão os atributos que essa tag suporta:

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
width	A largura da janela de exibição virtual do dispositivo. Valores1: largura do dispositivo ou a largura real em pixels, como 480
height	A altura da viewport virtual do dispositivo. Valores2: altura do dispositivo ou largura real em pixels, como 600
initial-scale	O zoom inicial quando a página é carregada. 1.0 não amplia.
minimun-scale	A quantia mínima que o visitante pode ampliar na página. 1.0 não amplia.
maximum-scale	A quantia máxima que o visitante pode ampliar na página. 1.0 não amplia.
user-scalable	Permite que o dispositivo aumente ou diminua o zoom. Os valores são sim ou não. Se definido como não, o usuário não poderá ampliar na página da web. O padrão é sim. As configurações do navegador podem ignorar esta regra.

Notas:

- 1 A propriedade width pode ser especificada em pixels (largura = 600) ou na largura do dispositivo (largura = largura do dispositivo) que representa a largura física da tela do dispositivo.
- 2 Da mesma forma, a propriedade height pode ser especificada em pixels (altura = 600) ou por altura do dispositivo (altura = altura do dispositivo) que representa a altura física da tela do dispositivo.

Refresh Automatico

Para atualizar a página a cada cinco segundos, adicione este elemento elemento no elemento principal:

```
<meta http-equiv="refresh" content="5" />
```

CUIDADO! Embora este seja um comando válido, é recomendável que você não o utilize devido aos seus efeitos negativos na experiência do usuário. Atualizar a página com muita frequência pode deixar de responder e, muitas vezes, rola até a Inicio da página. Se algumas informações da página precisarem ser atualizadas continuamente, existem maneiras muito melhores de faça isso atualizando apenas uma parte da página.

Reconhecimento de número de telefone

Plataformas móveis como o iOS reconhecem automaticamente os números de telefone e os transformam em links tel :. Embora o recurso seja muito prático, o sistema às vezes detecta códigos ISBN e outros números como números de telefone.

Para que o Safari móvel e alguns outros navegadores móveis baseados no WebKit desativem o reconhecimento e a formatação automáticos de números de telefone, você precisa desta metatag:

```
<meta http-equiv="format-detection" content="telephone=no" />
```

Redirect Automatico

Às vezes, sua página da web precisa de um redirecionamento automático. Por exemplo, para redirecionar para example.com após 5 segundos:

```
<meta http-equiv="refresh" content="5;url=https://tipsoce.com.br" />
```

Esta linha o enviará ao site designado (neste caso, exemplo.com, após 5 segundos.

Se você precisar alterar o atraso antes de um redirecionamento, basta alterar o número imediatamente antes do seu; url = will alterar o atraso de tempo.

Web App

Você pode configurar seu aplicativo Web ou site para adicionar um ícone de atalho ao aplicativo na tela inicial de um dispositivo e faça com que o aplicativo seja iniciado no "modo de aplicativo" em tela cheia usando o item de menu "Adicionar à tela inicial" do Chrome para Android.

Abaixo, as metatags abrem o aplicativo Web no modo de tela cheia (sem barra de endereço).

Android Chrome

```
<meta name="mobile-web-app-capable" content="yes" />
```

IOS

```
<meta name="apple-web-app-capable" content="yes" />
```

Você também pode definir a cor da barra de status e da barra de endereço na metatag.

Android Chrome

```
<meta name="theme-color" content="black" />
```

IOS

```
<meta name="apple-mobile-web-app-status-bar-style" content="black" />
```

Capítulo 28: Elementos de Códigos

<pre> e <code>

Se a formatação (espaço em branco, novas linhas, recuo) do código importa, use o elemento pre em combinação com

```
<pre>
    <code>
        x = 50
        if x == 50:
            print "É  50"
    </code>
</pre>
```

Você ainda precisa escapar de caracteres com significado especial em HTML (like < with <), assim, para exibir um bloco de HTML código (Este é um parágrafo.) poderia ficar assim:

```
<pre>
    <code>
        &lt; <p> Aqui vai um parágrafo</p>
    </code>
</pre>
```

<code> Inline

Se uma frase contiver código de computador (por exemplo, o nome de um elemento HTML), use o elemento de código para marcá-la acima:

```
<p> O <code> link </code> foi criado </p>
```

Capítulo 35: Citações

Em linha com elemento <q>

O elemento q pode ser usado para uma cotação que faz parte de uma frase:

```
<p> Ele falou <q>A resposta e 42.</q> e todos concordaram </p>
```

Aspas

Versão ≤ 4.01

As aspas não devem ser adicionadas. Os agentes do usuário devem (em HTML 4.01) resp. deve (em HTML 4.0) renderizá-los automaticamente.

Versão = 5

As aspas não devem ser adicionadas. Os agentes do usuário os renderizarão automaticamente.

URL de origem (atributo cite)

O **atributo cite** pode ser usado para referenciar o **URL** da fonte citada:

```
<p> Ele falou <q cite="https://tipscode.com.br/exemplo">A resposta e 42.</q> e todos concordaram </p>
```

Observe que os navegadores normalmente não mostram esse URL; portanto, se a fonte for relevante, você deve adicionar um hiperlink (um elemento) além do que, além do mais.

Em blocos com elemento <blockquote>

O elemento blockquote pode ser usado para uma citação (em nível de bloco):

```
<blockquote>
        </p> Opa tudo bem? </p>
</blockquote>
```

O atributo cite pode ser usado para referenciar o URL da fonte citada:

```
<blockquote cite="https://tipscode.com.br/atigos/ola-mundo">
        </p> Opa tudo bem? </p>
</blockquote>
```

Observe que os navegadores normalmente não mostram esse URL; portanto, se a fonte for relevante, você deve adicionar um hiperlink (um elemento) além disso (consulte a seção Citação / Atribuição sobre onde colocar este link).

Citação / Atribuição

Versão ≤ 4.01

A citação / atribuição não deve fazer parte do elemento blockquote:

```
<blockquote cite="https://tipscode.com.br/atigos/ola-mundo">
        </p> Opa tudo bem? </p>
</blockquote>
<p>Fonte: <cite><a href="https://exemplo.com" rel="external"</a></cite> Tudo sim</p>
```

Você pode adicionar um elemento div para agrupar a cotação e a citação, mas não existe maneira de associá-los semântica.

O elemento cite pode ser usado para a referência da fonte citada (mas não para o nome do autor).
Versão = 5

A citação / atribuição (por exemplo, o hiperlink que fornece o URL de origem) pode estar dentro da citação, mas nesse caso deve estar dentro de um elemento cite (para atribuições no texto) ou um elemento rodapé:

```
<blockquote cite="http://example.com/blog/hello-world">
      <p>The answer is 42.</p>
      <footer>
          <p>Source: <cite><a href="http://example.com/blog/hello-world" rel="external">Hello
          World</a></cite></p>
      </footer>
</blockquote>
```

O elemento cite pode ser usado para a referência da fonte citada ou para o nome do autor da citação.

Capítulo 36: Elemento <tabindex>

www.tipscode.com.br | Alisson Suassuna	
Atributos	Descrições
negative	O elemento será focalizável, mas não poderá ser alcançado através da navegação sequencial do teclado
0	O elemento será focalizável e alcançável através da navegação sequencial do teclado, mas sua ordem relativa é definida pela convenção da plataforma
positive	O elemento será focalizável e alcançável através da navegação sequencial do teclado, mas sua ordem relativa é definida pela convenção da plataforma

Adicionar um elemento à ordem de tabulação

```
<div tabindex="0">Alguns botões</div>
```

Nota: Tente usar um botão HTML nativo ou uma tag, quando apropriado.

Removendo elemento da tabulação

```
<button tabindex="-1"> Este botão não poderá ser acessado pela guia </div>
```

O elemento será removido da ordem de tabulação, mas ainda poderá ser focado.

Defina uma ordem de tabulação personalizada (não recomendo)

```
<div tabindex="2"> Segundo</div>
<div tabindex="1"> Primeiro</div>
```

Valores positivos inserirão o elemento na posição de ordem de tabulação de seu respectivo valor. Elementos sem A preferência (ou seja, tabindex = "o" ou elementos nativos como botão e a) será anexada após aqueles com preferência.

Valores positivos não são recomendados, pois perturbam o comportamento esperado das tabulações e podem confundir as pessoas que dependem de leitores de tela. Tente criar uma ordem natural reorganizando sua estrutura DOM.

Capítulo 37: Atributos Globais

Atributos	Descrições
class	Define um ou mais nomes de classe para um elemento. Consulte Classes e IDs
contenteditable	Define se o conteúdo de um elemento pode ser editado.
contextmenu	Define um menu de contexto mostrado quando um usuário clica com o botão direito do mouse em um elemento
dir	Define a direção do texto dentro de um elemento.
draggable	Define se um elemento pode ser arrastado.
hidden	Oculta um elemento não atualmente em uso na página.
id	Define um identificador exclusivo para um elemento. Consulte Classes e IDs.
lang	Define o idioma do conteúdo de um elemento e seus valores de atributo de texto. Consulte os idiomas do conteúdo
spellcheck	Define se a ortografia / gramática deve verificar o conteúdo de um elemento.
style	Define um conjunto de estilos CSS embutidos para um elemento
tabinddex	Define a ordem na qual os elementos em uma página são navegados pelo atalho do teclado da guia.
title	Define informações adicionais sobre um elemento, geralmente na forma de texto da dica de ferramenta ao passar o mouse.
translate	Define se é necessário traduzir o conteúdo de um elemento.

Atributo Editável por Conteúdo

```
<p contenteditable> Esse conteúdo é editável  </p>
```

Ao clicar no parágrafo, o conteúdo dele pode ser editado de forma semelhante a um campo de texto de entrada.

Quando o atributo contenteditable não está definido em um elemento, o elemento o herdará de seu pai. Então toda criança o texto de um elemento editável do conteúdo também será editável, mas você pode desativá-lo para um texto específico, como:

```
<p contenteditable>
        Esse conteúdo é editável
        <span contenteditable="false">  </span>
</p>
```

Observe que um elemento de texto não editável dentro de um elemento editável ainda terá um cursor de texto herdado de seu pai também.

Capítulo 38: HTML5 Cache

Exemplos básicos de cache

este é o nosso arquivo index.html

```
<!DOCTYPE html>
<html manifest="index.appcache">
<body>
    <p>Conteúdo</p>
</body>
</html>
```

então criaremos o arquivo index.appcache com os códigos abaixo

```
CACHE MANIFEST
index.html
```

escreva os arquivos que você deseja armazenar em cache, carregue index.html, vá para o modo offline e recarregue a guia

Nota: Os dois arquivos devem estar na mesma pasta neste exemplo

Capítulo 39: HTML Atributos de Eventos

Formulário de Eventos

Eventos acionados por ações em um formulário HTML (aplica-se a quase todos os elementos HTML, mas é mais usado no formulário elementos):

Atributos	Descrições
onblur	Dispara o momento em que o elemento perde o foco
onchange	Dispara o momento em que o valor do elemento é alterado
oncontextmenu	Script a ser executado quando um menu de contexto é acionado
onfocus	Dispara o momento em que o elemento fica em foco
oninput	Script a ser executado quando um elemento obtém entrada do usuário
oninvalid	Script a ser executado quando um elemento é inválido
onreset	Dispara quando o botão Redefinir em um formulário é clicado
onsearch	Dispara quando o usuário grava algo em um campo de pesquisa (para <input = "search">)
onselect	Dispara após a seleção de algum texto em um elemento
onsubmit	Dispara quando um formulário é enviado

Eventos de Teclados

Atributos	**Descrição**
onkeydown	Dispara quando um usuário está pressionando uma tecla
onkeypress	Dispara quando um usuário pressiona uma tecla
onkeyup	Dispara quando um usuário libera uma chave

Capítulo 40: Caracteres HTML

Símbolos e Caracteres

Muitos símbolos e caracteres especiais são necessários durante o desenvolvimento de uma página da web em html, mas como sabemos que às vezes o uso de caracteres diretamente pode interferir no código html real, que possui certos caracteres reservados e também não está disponível no teclado. Assim, para evitar o conflito e ao mesmo tempo para poder usar símbolos diferentes em nosso código, a w3 org nos fornece 'Entidades de caracteres'.

As entidades de caracteres são predefinidas com 'Nome da entidade' - & entity_name; e 'Número da entidade' - & entity_number; então nós precisamos usar um dos dois para que o símbolo necessário seja renderizado em nossa página.

A lista de poucas entidades de caracteres pode ser encontrada em https://dev.w3.org/html5/html-author/charref

Um exemplo simples com o uso da entidade de caractere para 'lupa':

```
<input type="text" placeholder="&#128269; Seatch" />
```

Resultado:

Caracteres Especiais comuns

Alguns caracteres podem ser reservados para HTML e não podem ser usados diretamente, pois podem obstruir os códigos HTML reais. Por exemplo, tentar exibir os colchetes dos ângulos esquerdo e direito (<>) no código-fonte pode causar problemas inesperados resulta na saída. Da mesma forma, os espaços em branco escritos no código-fonte podem não ser exibidos conforme o esperado no HTML de saída. Alguns, como ☎, não estão disponíveis no conjunto de caracteres ASCII.

Para esse fim, são criadas entidades de caracteres. Estes são da forma & entity_name; ou & entity_number ;. o A seguir, estão algumas das entidades HTML disponíveis.

www.tipscode.com.br | Alisson Suassuna			
Caractere	Descrições	Nome da Entidade	Número da Entidade
" "	non-breakink space		
"<"	less than	&lt;	&#60;
">"	greater than	&gt;	&#62;
"&"	ampersand	&amp;	&#38;
"—"	em dash	&mdash;	&#8212;
"–"	em dash	&ndash;	&#8211;
"©"	copyright	&copy;	&#169;
"®"	registered trademark	&reg;	&#174;
"™"	trademark	&trade;	&#8482;
"☎"	phone	&phone;	&#9742;

o seguinte código HTML é usado:

```
<strong>&copy: TipsCode.com.br </strong>
```

Conheça Aqui