

VIEW

# HTML: FALANDO A LÍNGUA DA INTERNET

HENRIQUE R. POYATOS NETO



03

**LISTA DE FIGURAS**

Figura 1 – Aba exibindo o título do documento .....	12
Figura 2 – Documento HTML com ícone, um dos usados da <i>tag</i> <link> .....	14
Figura 3 – <i>Tag</i> <style> em funcionamento .....	17
Figura 4 – <i>Tag</i> <p> em funcionamento .....	20
Figura 5 – <i>Tag</i>   em funcionamento .....	21
Figura 6 – <i>Tags</i> <h1>, <h2> e <h3> em funcionamento .....	22
Figura 7 – <i>Tag</i> <img> em dois momentos: com a imagem origem não localizada e com a imagem localizada .....	24
Figura 8 – <i>Tag</i> <div> em funcionamento .....	25
Figura 9 – <i>Tag</i> <a> em funcionamento .....	28
Figura 10 – Hyperlink aplicado em uma imagem .....	29
Figura 11 – Lista ordenada .....	30
Figura 12 – Lista não ordenada.....	31
Figura 13 – <i>iframe</i> em dois momentos: na carga da página e ao clicar no <i>hyperlink</i> disponível.....	33
Figura 14 – Tabela criada .....	36
Figura 15 – Tabela completa.....	39
Figura 16 – Dimensionando a primeira célula (e coluna) em 50% do tamanho total da tabela .....	41
Figura 17 – Mesclando células horizontalmente .....	42
Figura 18 – Mesclando células verticalmente.....	43
Figura 19 – Rótulo para campos de formulário .....	46
Figura 20 – Exemplo de formulário usando <input> com <i>value</i> e <i>disabled</i> .....	50
Figura 21 – Exemplo de formulário usando <input> com <i>checked</i> .....	51
Figura 22 – Exemplo de formulário usando <input> .....	52
Figura 23 – Exemplo de formulário usando <select> e <option> com opção previamente selecionada .....	54
Figura 24 – Exemplo de formulário usando <select>, <option> e <optgroup> com seleção múltipla.....	55
Figura 25 – Exemplo de formulário usando <textarea>.....	56
Figura 26 – Exemplo de formulário usando <fieldset> e <legend> .....	57
Figura 27 – Exemplo de formulário usando <datalist> em dois momentos: como caixa de seleção e digitando uma opção não encontrada na lista .....	58
Figura 28 – Exemplo de formulário usando <progress>.....	59

## LISTA DE CÓDIGOS-FONTE

Código-fonte 1 – Padrão de codificação HTML, derivado de SGML .....	8
Código-fonte 2 – Tag <code>&lt;!DOCTYPE&gt;</code> no padrão HTML 5 .....	9
Código-fonte 3 – Tag <code>&lt;!DOCTYPE&gt;</code> no padrão XHTML 1.0 .....	9
Código-fonte 4 – Tag <code>&lt;html&gt;</code> no padrão XHTML 1.0 .....	9
Código-fonte 5 – Tag <code>&lt;html&gt;</code> no padrão HTML 5 .....	9
Código-fonte 6 – Tag <code>&lt;head&gt;</code> .....	11
Código-fonte 7 – Tag <code>&lt;title&gt;</code> .....	11
Código-fonte 8 – Tag <code>&lt;link&gt;</code> vinculando arquivos CSS no padrão HTML 5 .....	12
Código-fonte 9 – Tag <code>&lt;link&gt;</code> vinculando arquivos CSS no padrão XHTML 1.0 .....	13
Código-fonte 10 – Tag <code>&lt;link&gt;</code> vinculando arquivo de ícone no padrão HTML 5 .....	14
Código-fonte 11 – Tag <code>&lt;meta&gt;</code> sendo utilizada para definir o padrão de codificação de caracteres .....	15
Código-fonte 12 – Tag <code>&lt;meta&gt;</code> sendo utilizada para recarregar o documento .....	15
Código-fonte 13 – Tag <code>&lt;meta&gt;</code> sendo utilizada para definir as palavras-chave .....	16
Código-fonte 14 – Tag <code>&lt;style&gt;</code> .....	17
Código-fonte 15 – Tag <code>&lt;script&gt;</code> com código JavaScript embutido no HTML .....	18
Código-fonte 16 – Tag <code>&lt;script&gt;</code> com referência a um arquivo externo .....	18
Código-fonte 17 – Tag <code>&lt;body&gt;</code> .....	19
Código-fonte 18 – Tag <code>&lt;p&gt;</code> aplicada ao HTML .....	19
Código-fonte 19 – Tag <code>&lt;br&gt;</code> aplicada ao HTML .....	20
Código-fonte 20 – Uso das tags <code>&lt;h1&gt;</code> , <code>&lt;h2&gt;</code> e <code>&lt;h3&gt;</code> aplicadas ao HTML .....	21
Código-fonte 21 – Tag <code>&lt;img&gt;</code> aplicada ao HTML .....	23
Código-fonte 22 – Tag <code>&lt;div&gt;</code> aplicada ao HTML .....	24
Código-fonte 23 – Exemplo anterior trocando <code>&lt;div&gt;</code> por <code>&lt;article&gt;</code> .....	26
Código-fonte 24 – Exemplo de uso das novas tags semânticas .....	27
Código-fonte 25 – Tag <code>&lt;a&gt;</code> implementada em HTML .....	28
Código-fonte 26 – Tags <code>&lt;a&gt;</code> e <code>&lt;img&gt;</code> criando um <i>hyperlink</i> em uma imagem .....	29
Código-fonte 27 – Tags <code>&lt;ol&gt;</code> e <code>&lt;li&gt;</code> criando uma lista ordenada .....	30
Código-fonte 28 – Tags <code>&lt;ul&gt;</code> e <code>&lt;li&gt;</code> criando uma lista não ordenada .....	30
Código-fonte 29 – O arquivo <code>iframe_pagina1.html</code> .....	31
Código-fonte 30 – O arquivo <code>iframe_pagina2.html</code> .....	32
Código-fonte 31 – O arquivo <code>iframe.html</code> .....	32
Código-fonte 32 – Tag <code>&lt;table&gt;</code> sendo utilizada .....	35
Código-fonte 33 – Tags <code>&lt;thead&gt;</code> , <code>&lt;tbody&gt;</code> e <code>&lt;tfoot&gt;</code> sendo utilizadas .....	36
Código-fonte 34 – Tags <code>&lt;thead&gt;</code> , <code>&lt;tbody&gt;</code> e <code>&lt;tfoot&gt;</code> sem suas tags de finalização ..	37
Código-fonte 35 – Linhas de tabela sendo definidas com a tag <code>&lt;tr&gt;</code> .....	38
Código-fonte 36 – Exemplo de tabelas usando <code>&lt;td&gt;</code> e <code>&lt;th&gt;</code> .....	39
Código-fonte 37 – Mudando as dimensões de uma célula (e coluna) .....	40
Código-fonte 38 – Mesclando células horizontalmente .....	42
Código-fonte 39 – Mesclando células verticalmente .....	43
Código-fonte 40 – Exemplo de aplicação da tag <code>&lt;form&gt;</code> em HTML .....	45
Código-fonte 41 – Exemplo de uso da tag <code>&lt;label&gt;</code> em HTML .....	46
Código-fonte 42 – Exemplo de formulário usando <code>&lt;input&gt;</code> com <i>value</i> e <i>disabled</i> ....	49
Código-fonte 43 – Exemplo de formulário usando <code>&lt;input&gt;</code> com <i>checked</i> .....	50
Código-fonte 44 – Exemplo de formulário usando <code>&lt;input&gt;</code> .....	52
Código-fonte 45 – Exemplo de formulário usando <code>&lt;select&gt;</code> e <code>&lt;option&gt;</code> com opção previamente selecionada .....	53

Código-fonte 46 – Exemplo de formulário usando <select>, <option> e <optgroup> com seleção múltipla .....	54
Código-fonte 47 – Exemplo de formulário usando <textarea> .....	56
Código-fonte 48 – Exemplo de formulário usando <fieldset> e <legend> .....	56
Código-fonte 49 – Exemplo de formulário usando <datalist> .....	58
Código-fonte 50 – Exemplo de formulário usando <progress> .....	59

EXEMPLO

## SUMÁRIO

1 HTML: FALANDO A LÍNGUA DA INTERNET .....	6
1.1 Introdução .....	6
1.2 Sobre a linguagem HTML.....	6
1.3 Considerações sobre os diferentes padrões .....	6
1.4 Padrão de codificação .....	7
1.5 Tags Essenciais .....	8
1.5.1 TAG <!DOCTYPE> .....	8
1.5.2 TAG <HTML> .....	9
1.5.3 Tag <head> .....	11
1.5.4 Tag <title> .....	11
1.5.5 Tag <link>.....	12
1.5.6 Tag <meta> .....	14
1.5.7 Tag <style>.....	16
1.5.8 Tag <script> .....	17
1.5.9 Tag <body> .....	18
1.6 Tags Básicas.....	19
1.6.1 TAG <P> .....	19
1.6.2 TAG   .....	20
1.6.3 Tags <h1>, <h2>, <h3>...até <h6> .....	21
1.6.4 Tag <img> .....	22
1.6.5 Tag <div> .....	24
1.6.6 SUBSTITUTIVOS DO <DIV> PARA MELHORAR A SEMÂNTICA .....	25
1.6.7 TAG <A> .....	27
1.6.8 TAGS <OL>, <UL> E <LI> .....	29
1.6.9 TAG <IFRAME> .....	31
1.6.10 Tags <frame> e <frameset>: por que não usar? .....	33
1.6.11 TAGS DE MARCAÇÃO A SE ESQUECER.....	34
2 TABULAÇÃO DE DADOS: CRIANDO TABELAS .....	35
2.1 Tag <table> .....	35
2.2 Tags <thead>, <tbody> e <tfoot> .....	36
2.3 Tag <tr>.....	37
2.4 Tags <td> e <th> .....	38
2.5 Dimensionando células .....	40
2.6 Mesclando colunas.....	41
3 CRIANDO FORMULÁRIOS.....	44
3.1 Tag <form> .....	44
3.2 Tag <label> .....	46
3.3 Tag <input> .....	47
3.4 Tags <select>, <option> e <optgroup>.....	52
3.5 Tag <textarea> .....	55
3.6 Tags <fieldset> e <legend> .....	56
3.7 Tag <datalist>.....	57
3.8 Tag <progress> .....	58
CONCLUSÃO.....	60
REFERÊNCIAS .....	61

# 1 HTML: FALANDO A LÍNGUA DA INTERNET

## 1.1 Introdução

Conhecer HTML é indispensável atualmente. Mesmo que o desenvolvedor não tenha qualquer interesse em desenvolvimento web, a linguagem de marcação de hipertexto se tornou padrão em outras plataformas, atingindo aplicações desktop e até mesmo os badalados apps dos *smartphones*.

É com este conhecimento que o Health Track ganhará uma “cara”. Embarque conosco nesta busca pelo conhecimento!

## 1.2 Sobre a linguagem HTML

A linguagem de marcação de hipertexto HTML foi criada com o objetivo de diagramar documentos para o serviço de *World Wide Web* (WWW). Em seus primórdios, foi concebida pelo físico britânico Tim Berners-Lee para reproduzir documentos de pesquisa e compartilhá-los com facilidade. O padrão logo foi adotado pelas universidades que começavam a fazer parte da Internet, na década de 1990, na reprodução e compartilhamento de teses acadêmicas.

A WWW, no entanto, evoluiu muito nas últimas décadas: ela é usada de forma comercial, para divulgar e vender produtos e serviços de empresas; estreitando os relacionamentos e comunicação em redes sociais; e no entretenimento, veiculando áudio, vídeo e jogos. Uma boa linguagem, seja ela de programação ou marcação, é realmente boa quando permanece viva, ou seja, é constantemente atualizada para se adequar à realidade atual. Felizmente, o HTML é uma dessas linguagens.

## 1.3 Considerações sobre os diferentes padrões

A evolução mencionada gerou várias especificações diferentes. Muitas páginas na Internet estão escritas no padrão HTML 4.01, publicado em 24 de dezembro de 1999. Uma reformulação do HTML 4 foi lançada quase um mês depois, e ficou conhecida como *Extensible Hypertext Markup Language*, o XML 1.0, reformulado em

1º de agosto de 2002. E, por fim, outras páginas utilizam a revisão, o XML 1.1, cuja segunda versão foi liberada em 16 de agosto de 2006.

Concentraremos nossos estudos no último padrão vigente, o HTML 5, que tem sido trabalhado intensamente nos últimos anos e foi considerado “recomendado para uso” pela W3C, em 28 de outubro de 2014. Explico o porquê: o padrão foi rapidamente adotado por todos os fabricantes de navegadores do mercado, incluindo os usados em dispositivos portáteis.

De acordo com o site W3Counter em números de dezembro de 2015, aproximadamente 70% utilizam versões de navegadores extremamente atualizados (Google Chrome em versões 38, 46 e 47, Mozilla Firefox 42 e 43, Internet Explorer 11, Safari versões 8 e 9, entre outros). O padrão HTML 5 seria um grande problema para navegadores Internet Explorer nas versões 8 para baixo. Pois bem, sendo a mesma fonte, apenas 1,9% dos visitantes está utilizando IE nas versões 6, 7 e 8.

Sendo assim, utilize o padrão HTML 5 em novos documentos HTML. Exceções devem ser feitas para um ambiente Intranet cuja empresa possui um parque tecnológico extremamente defasado; ou aplicações web desenvolvidas com foco em especificidades do IE 6, incompatíveis com as versões posteriores, forçando gestores de TI a proibir atualizações de navegadores web (obrigado, Microsoft!). Nesses casos, recomenda-se o padrão XML 1.0.

Com o lançamento e a consolidação do Microsoft Edge como principal navegador web da Microsoft, esse cenário teve uma grande melhora para os desenvolvedores que agora podem trabalhar com o que há de mais moderno nas especificações de tecnologias web.

#### 1.4 Padrão de codificação

A linguagem HTML deriva da metalinguagem *Standard Generalized Markup Language* (SGML), inventada pela IBM da década de 1960. As sintaxes de marcação são informadas entre “<” (sinal de menor) e “>” (sinal de maior) e, a partir de agora, serão chamadas de *tags*. Essas, por sua vez, possuem uma hierarquia, ou seja, existem “*subtags*” que só devem ser posicionadas dentro de uma determinada *tag*

“mãe”. Por essa razão, a maioria das *tags* HTML é aberta e fechada (o fechamento usa o símbolo “/”, barra).

Veja o exemplo:

```
<tag>
  <subtag  atributo="valor">Informação  cuja  marcação
atuará</subtag>
</tag>
```

Código-fonte 1 – Padrão de codificação HTML, derivado de SGML  
Fonte: Elaborado pelo autor (2016)

As *tags* desse exemplo são fictícias, mas dão uma boa ideia do que virá pela frente. *Tags* possuem atributos que possibilitam a parametrização do efeito gerado pela *tag* e seguem o padrão exibido no exemplo: atributo=“valor”.

## 1.5 Tags essenciais

Vamos começar com as *tags* essenciais, aquelas que todo documento HTML deve possuir.

### 1.5.1 TAG <!DOCTYPE>

O *Document Type Definition* (DTD) ou a *tag* <!DOCTYPE> não é exatamente parte do padrão HTML, mas é muito importante. Conforme dito anteriormente, existem vários padrões HTML. Pois bem, como o navegador do usuário sabe em qual desses padrões do documento HTML foi escrito? Bem, ele não sabe, ele “adivinha” qual é e nem sempre acerta.

Trata-se da primeira *tag* de um documento colocada antes mesmo da <html>. Existem dezenas de *doctype*s diferentes que, ao serem interpretadas pelo navegador, não só determinam o padrão a ser seguido como ativam modos de renderização diferentes: *standards mode* é seu modo peculiar de renderização; *quirks mode* é utilizado para páginas antigas; ou o modo intermediário quase padronizado conhecido como *almost standards mode* (sei que parece piada, mas não é).

Para o padrão HTML 5, a *tag* foi devidamente encurtada, não exigindo a especificação DTD determinada. A linha de código fica assim:



```
<!DOCTYPE html>
```

Código-fonte 2 – *Tag <!DOCTYPE>* no padrão HTML 5  
Fonte: Elaborado pelo autor (2016)

Caso o documento seja escrito em XHTML 1.0, a *tag <!DOCTYPE>* fica assim:

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
```

Código-fonte 3 – *Tag <!DOCTYPE>* no padrão XHTML 1.0  
Fonte: Elaborado pelo autor (2016)

### 1.5.2 TAG <HTML>

A *tag <html>* determina onde começa e termina o documento HTML. Todas as outras *tags* devem ser contidas dentro dela. Em XHTML 1.0, a declaração era:

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
lang="pt-br" xml:lang="pt-br">
</html>
```

Código-fonte 4 – *Tag <html>* no padrão XHTML 1.0  
Fonte: Elaborado pelo autor (2016)

No padrão HTML 5, os atributos “xmlns” e “xml:lang” não são mais necessários, e a linha pode ser reduzida para:

```
<!DOCTYPE html>
<html lang="pt-br">
</html>
```

Código-fonte 5 – *Tag <html>* no padrão HTML 5  
Fonte: Elaborado pelo autor (2016)

O atributo *lang* é global, por isso, pode ser utilizado em qualquer *tag* da linguagem HTML. Entretanto, é mais comum que seja informado em *<html>*, determinando assim o idioma do documento todo.

A importância é vital: como acontece com **<!DOCTYPE>**, se o idioma não for indicado aqui, o navegador tenta adivinhar qual é, podendo falhar miseravelmente. Além disso, mecanismos de busca se beneficiam dessa indicação de idioma, tornando-se, assim, mais eficazes. Algoritmos do Google dão preferência a buscas

orgânicas para documentos cujo **lang** é utilizado, ou seja, é absolutamente obrigatório. Falaremos muito aqui sobre essas boas práticas relacionadas ao SEO (*Search Engine Optimization*).

### Atributos globais

Conforme dito, atributos globais são aqueles que podem ser aplicados em qualquer elemento/tag HTML. A seguir, relacionaremos os principais:

- **accesskey**: com ele, é possível determinar um teclado de atalho para o elemento; ao digitar usando a tecla aqui informada, o elemento HTML em questão ganha foco.
- **class**: este atributo está relacionado com o CSS, de que falaremos adiante. Deve ser usado para informar a classe que será utilizada para estilizar o elemento em questão (exemplos no capítulo de CSS).
- **id**: utilizado para informar ou identificar um único elemento. Como o nome já indica, ele deve ser único, nenhum outro elemento poderá ter o mesmo id, que pode ser uma palavra ou uma palavra seguida de número. Ele segue basicamente as mesmas regras de batismo a que estamos acostumados ao declarar variáveis em linguagens de programação. Será muito útil quando tratarmos de JavaScript (em um capítulo posterior).
- **lang**: determina o idioma utilizado dentro do elemento HTML.
- **style**: pode ser usado para aplicar estilos no padrão CSS diretamente no elemento. Sendo assim, o valor dentro desse atributo é linguagem CSS pura. Mais uma dica de SEO aqui: style não é a solução ideal, prefira usar o atributo class.
- **tabindex**: por questões de rapidez (ou pela falta do mouse), alguns usuários gostam de navegar entre os elementos HTML usando a tecla TAB do teclado; aliás, essa forma de navegação é muito utilizada por deficientes visuais. Conforme a tecla TAB é acionada, um novo elemento do HTML ganha foco. Pois bem: nem sempre a ordem estabelecida é a mais adequada. Você pode modificar isso, atribuindo números (começando em 0, sendo esse o primeiro elemento do documento a ganhar foco), assim, a navegação por TAB pode ser inteiramente personalizada.

- **title:** representa informações do elemento, como uma dica (ou tooltip). No caso de um hyperlink, pode ser o título ou uma descrição do destino; se for uma imagem, utilizar seu crédito ou uma descrição da imagem (essencial para leitores de tela de deficientes visuais); e no caso de um parágrafo, usar uma nota de rodapé ou comentário sobre o texto.

Não se preocupe, veremos adiante exemplos utilizando os atributos globais aqui descritos.

### 1.5.3 Tag <head>

Determina o cabeçalho de um documento HTML. Serve, na verdade, como um contêiner de outras *tags* importantes para o documento que não faz parte do corpo deste propriamente dito:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
  </head>
</html>
```

Código-fonte 6 – Tag <head>  
Fonte: Elaborado pelo autor (2016)

### 1.5.4 Tag <title>

Utilizada para informar o título do documento HTML. Como <title> faz parte do cabeçalho (<head>), esse título não é renderizado em tela – somente as *tags* que fazem parte do corpo do documento o serão. Entretanto, não é por isso que não é importante: trata-se do título que será utilizado pelos mecanismos de buscas, topo e abas do seu navegador.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de Título</title>
  </head>
</html>
```

Código-fonte 7 – Tag <title>  
Fonte: Elaborado pelo autor (2016)

Esse código-fonte produz o seguinte resultado:

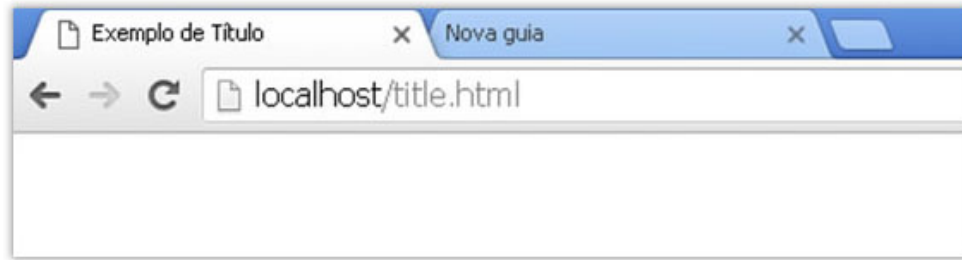


Figura 1 – Aba exibindo o título do documento  
Fonte: Elaborado pelo autor (2016), adaptado por FIAP (2017)

Ufa! Finalmente consegui lhe mostrar algo, não é mesmo? Calma, temos muito pela frente e várias coisas interessantes para ver.

### 1.5.5 Tag <link>

<link> não é a *tag* dos tradicionais *hyperlinks* clicáveis da Internet: quem faz isso é a *tag* <a>, que veremos mais adiante. Lembre-se: nem começamos a abordar o corpo do documento, essa *tag* é filha de <head>.

Para que serve <link>, então? Vincular documentos. Uma página muitas vezes é composta de vários arquivos: mantemos o HTML, o CSS e o JavaScript em arquivos separados (essa é a boa prática e, portanto, prática SEO).

Ela é utilizada comumente para vincular os arquivos CSS necessários. Veja o exemplo abaixo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de Título</title>
    <link      rel="stylesheet"      type="text/css"
href="css/principal.css" title="Estilos principais">
    <link      rel="stylesheet"      type="text/css"
href="css/títulos.css" title="Títulos">
  </head>
</html>
```

Código-fonte 8 – Tag <link> vinculando arquivos CSS no padrão HTML 5  
Fonte: Elaborado pelo autor (2016)

É interessante notar que a *tag* <link> não possui uma *tag* de fechamento (como </link>). Isso acontece porque ela não está marcando um trecho de texto, não existe conteúdo a ser contido.

Pelo padrão HTML 5, a *tag* é simplesmente aberta e não é fechada, como no exemplo acima. Pelo padrão XHTML, a *tag* abre e fecha em si mesma, como pode ser visto no exemplo a seguir:

```
<!DOCTYPE html
    PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    lang="pt-br" xml:lang="pt-br">
  <head>
    <title>Exemplo de Título</title>
    <link      rel="stylesheet"          type="text/css"
href="css/principal.css" title="Estilos principais" />
    <link      rel="stylesheet"          type="text/css"
href="css/títulos.css" title="Títulos" />
  </head>
</html>
```

Código-fonte 9 – Tag <link> vinculando arquivos CSS no padrão XHTML 1.0

Fonte: Elaborado pelo autor (2016)

Falemos agora dos principais atributos específicos de <link>:

- **href:** abreviação de *hyperlink reference*, trata-se do endereço do arquivo que está sendo vinculado. O endereço pode ser absoluto (caso estejamos acessando um arquivo externo, como `http://www.outrosite.com.br/css/estilos.css`) ou relativo. Um caminho relativo toma como referência o diretório em que o documento HTML está como base. Por exemplo, `href="arquivo.css"` significa que o arquivo CSS está no mesmo diretório que o documento HTML que lhe faz referência. Em nosso exemplo, os dois arquivos estão em uma subpasta chamada "css" (boa prática!).
- **rel:** qual o relacionamento entre os documentos vinculados? Existem várias possibilidades: pode estar vinculado a um *stylesheet* (folha de estilo, no caso, CSS) ou a um *icon* (ícone), por exemplo. Vários outros valores foram criados para a versão HTML 5 (vide o material/curso específico de HTML 5).
- **type:** qual o tipo MIME de documento vinculado? No caso de CSS, o valor deve ser "text/css".

Que tal um exemplo vinculado a um arquivo para um ícone deste documento? Embora os navegadores modernos aceitem ícones de imagens em formato PNG, usaremos o padrão, que são ícones no formato ICO:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de Título</title>
    <!-- IE, sempre você... -->
    <link rel="shortcut icon" type="image/x-icon"
href="imagens/html5.ico" />
    <!-- todos os outros navegadores -->
    <link rel="icon" type="image/x-icon"
href="imagens/html5.ico" />
  </head>
</html>
```

Código-fonte 10 – Tag <link> vinculando arquivo de ícone no padrão HTML 5  
Fonte: Elaborado pelo autor (2016)

Essa é uma das ocasiões em que o Internet Explorer diverge do padrão: ele espera que o *rel* informado seja “*shortcut icon*”, enquanto todos os outros navegadores esperam o valor “*icon*”.

A propósito, isso é uma *tag* de <!-- Comentário -->.



Figura 2 – Documento HTML com ícone, um dos usos da tag <link>  
Fonte: Elaborado pelo autor (2016), adaptado por FIAP (2017)

### 1.5.6 Tag <meta>

Este elemento é utilizado para informar os metadados do documento HTML. Trata-se de uma *tag* que também não possui fechamento (</meta>), assim como <link>. Possui várias aplicações:

**Padrão de codificação de caracteres:** existem dezenas de padrões de codificação de caracteres, pois temos diversos alfabetos e padrões diferentes de escrita no mundo todo. O padrão utilizado no Brasil é o “iso-8859-1”, conhecido em outros meios como “latin1”.

Existe, no entanto, um padrão unificado, conhecido como “utf-8”, que tem sido amplamente aceito pelos principais portais brasileiros. Recomendo o uso desse padrão, mas tenha o cuidado de salvar o código-fonte exatamente no mesmo padrão em sua suíte de desenvolvimento (sim, arquivos de texto puro também possuem os mesmos padrões de codificação de caracteres).

Não usar a *tag* <meta> para informar o padrão de codificação fará com que o navegador procure adivinhá-lo. E se tem uma coisa com o potencial de dar realmente errado, é isso. Informá-lo é essencial e faz parte das práticas SEO.

O atributo *charset* deve ser utilizado para esse fim, como pode ser visto no exemplo a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo meta</title>
    <meta charset="utf-8">
  </head>
</html>
```

Código-fonte 11 – Tag <meta> sendo utilizada para definir o padrão de codificação de caracteres  
Fonte: Elaborado pelo autor (2016)

**Recarregar a página html:** usando os atributos http-equiv e content, podemos definir a rotina de recarga automática da página. O valor de http-equiv será "Refresh" e o content armazena o número de segundos para a recarga. O exemplo a seguir recarrega a página após 5 minutos:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo meta </title>
    <meta http-equiv="Refresh" content="300">
  </head>
</html>
```

Código-fonte 12 – Tag <meta> sendo utilizada para recarregar o documento  
Fonte: Elaborado pelo autor (2016)

**Palavras-chave do documento:** com os atributos name e content, podemos utilizar <meta> para armazenar as palavras-chave do conteúdo. Informá-las é importante do ponto de vista SEO, mas é interessante ressaltar que essa *tag* é apenas um dos fatores analisados pelos motores de busca.

Veja o exemplo a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo meta</title>
    <meta name="keywords" content="fiap, exemplo, html">
  </head>
</html>
```

Código-fonte 13 – Tag <meta> sendo utilizada para definir as palavras-chave  
Fonte: Elaborado pelo autor (2016)

### 1.5.7 Tag <style>

Permite aplicar estilos CSS diretamente no documento. A boa prática é, no entanto, criar um arquivo .css separado e vinculá-lo com a *tag* <link>.

Veja o código-fonte do exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de Título</title>
    <meta charset="utf-8">
    <meta name="keywords" content="fiap, exemplo, html">
    <style>
      body { background-color: red; }
    </style>
  </head>
  <body>
```



```
Conteúdo

</body>

</html>
```

Código-fonte 14 – Tag <style>  
Fonte: Elaborado pelo autor (2021)

Agora este em funcionamento:

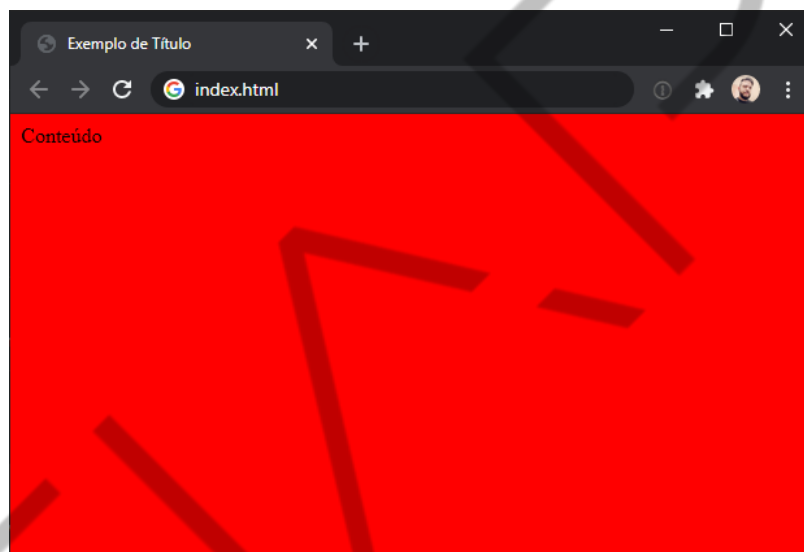


Figura 3 – Tag <style> em funcionamento  
Fonte: Elaborado pelo autor (2016), adaptado por FIAP (2017)

### 1.5.8 Tag <script>

Utilizada para inserir comandos de linguagem do tipo *script* que sejam interpretados pelo navegador (ou seja, do lado cliente). Resumidamente, JavaScript.

Os comandos da linguagem ficam entre as *tags* <script>, definindo a linguagem (“text/javascript”) e utilizando o atributo “type”. Isso era necessário, pois, no passado, exibiam outras opções de linguagem do tipo *script*, como o VBScript.

Recentemente, a definição do “type” se tornou opcional para o JavaScript.

Veja um exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo JavaScript</title>
  </head>
  <body>
  </body>
</html>
<script type="text/javascript">
  window.alert("Olá, mundo!");
</script>
```

Código-fonte 15 – Tag <script> com código JavaScript embutido no HTML  
Fonte: Elaborado pelo autor (2016)

Entretanto, essa não é a melhor abordagem, pois os códigos da linguagem *JavaScript* ficam misturados com a linguagem de marcação HTML. Além de dificultar a manutenção, essa prática não possibilita o reaproveitamento de código, compartilhando as instruções *JavaScript* com outros documentos HTML (aliás, este raciocínio é aplicável ao caso do CSS).

A boa prática é separá-los, criando um arquivo de extensão .js à parte e vinculá-lo ao documento HTML, utilizando o atributo “src” (*source*):

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo JavaScript</title>
    <script src="arquivo.js"></script>
  </head>
  <body>
  </body>
</html>
```

Código-fonte 16 – Tag <script> com referência a um arquivo externo  
Fonte: Elaborado pelo autor (2016)

Falaremos bastante sobre o assunto em um capítulo posterior.

### 1.5.9 Tag <body>

Determina o corpo de um documento HTML. Todas as *tags* a partir daqui ficarão contidas dentro desse elemento.

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de body</title>
    <meta charset="utf-8">
  </head>
  <body>
  </body>
</html>
```

Código-fonte 17 – Tag <body>  
Fonte: Elaborado pelo autor (2016)

## 1.6 Tags básicas

Seguimos agora para as *tags* básicas, abordando as *tags* de agrupamento e marcação de texto.

### 1.6.1 TAG <P>

Utilizada para definir o início e o fim de um parágrafo.

Repare que os navegadores atribuem automaticamente uma margem entre um parágrafo e outro, que pode ser eliminada ou até mesmo aumentada usando CSS.

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de Parágrafo</title>
    <meta charset="utf-8">
  </head>
  <body>
    <p>Primeiro parágrafo</p>
    <p>Segundo parágrafo</p>
  </body>
</html>
```

Código-fonte 18 – Tag <p> aplicada ao HTML  
Fonte: Elaborado pelo autor (2016)

Primeiro parágrafo  
Segundo parágrafo

Figura 4 – Tag <p> em funcionamento  
Fonte: Elaborado pelo autor (2016)

### 1.6.2 TAG <BR>

Vem da palavra *breakline*, ou seja, quebra de linha, utilizada antes para “pular” de linha. Essa *tag* **não** possui versão de fechamento (</br>).

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de quebra de linha</title>
    <meta charset="utf-8">
  </head>
  <body>
    Linha 1<br>
    Linha 2<br>
    Linha3
  </body>
</html>
```

Código-fonte 19 – Tag <br> aplicada ao HTML  
Fonte: Elaborado pelo autor (2016)

Linha 1  
Linha 2  
Linha 3

Figura 5 – Tag <br> em funcionamento  
Fonte: Elaborado pelo autor (2016)

### 1.6.3 Tags <h1>, <h2>, <h3>... até <h6>

Trata-se de *tags* específicas para títulos e subtítulos, sendo possível, portanto, subdividir conteúdos até um sexto nível.

Veja o exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo h1 até h6</title>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>1. Título</h1>
    <p> Introdução do capítulo</p>

    <h2>1.1. Subtítulo</h2>
    <p> Introdução desta seção</p>

    <h3>1.1.1. Subtítulo</h3>
    <p> Introdução desta subseção</p>
  </body>
</html>
```

Código-fonte 20 – Uso das *tags* <h1>, <h2> e <h3> aplicadas ao HTML  
Fonte: Elaborado pelo autor (2016)

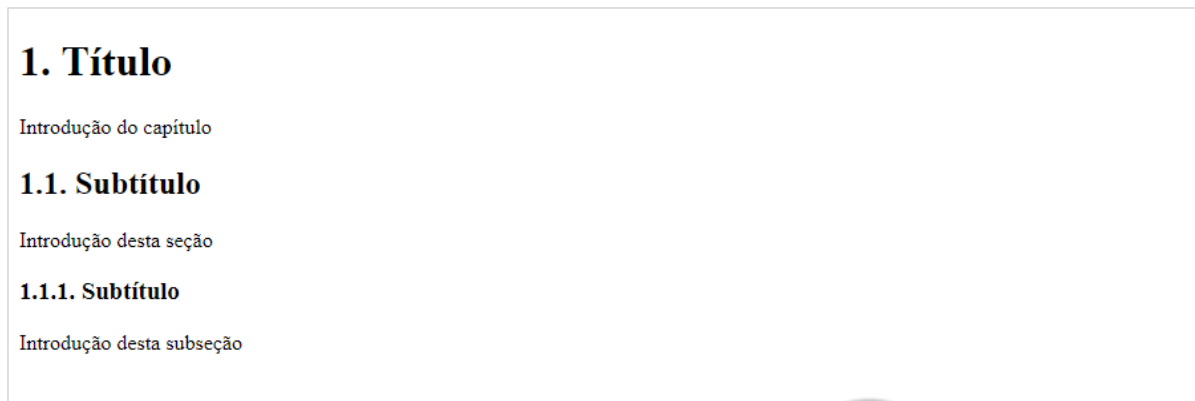


Figura 6 – Tags `<h1>`, `<h2>` e `<h3>` em funcionamento  
Fonte: Elaborado pelo autor (2016), adaptado por FIAP (2017)

Repare que o tamanho e os estilos das *tags* `<h1>` até `<h6>` são diferentes, `<h1>` possuindo um tamanho maior que o `<h2>`, que tem um tamanho maior que `<h3>` e assim por diante.

Muitos diagramadores HTML acabam usando, por exemplo, `<h3>` como título, sem que haja um `<h2>` e `<h1>` em seu documento, julgando o tamanho dos anteriores grande demais para seu documento. Trata-se de um equívoco: existe uma hierarquia entre essas *tags*. Se o documento possui um `<h4>`, significa que temos antes dele `<h1>` até `<h3>`. Se, por acaso, a estilização-padrão de `<h1>` for grande demais para você, altere-a usando CSS (veremos como fazer isso adiante).

#### 1.6.4 Tag `<img>`

É usada para aplicar imagens ao documento HTML. Para isso, conta com seus principais atributos:

**src:** utilizado para informar o caminho da imagem. Pode ser um caminho relativo, como explicado antes, ou um caminho absoluto, apontando até mesmo para imagens que estejam em outro domínio de Internet. Essa abordagem não é muito recomendada, por várias razões: você possui autorização de uso dessa imagem? Além disso, mesmo que haja uma autorização, como ter certeza de que a imagem estará sempre disponível (e não seja apagada) já que não se tem o controle deste domínio? É sempre recomendado, portanto, que as imagens estejam hospedadas no mesmo local que os documentos HTML em geral.

**width e height:** trata-se da largura e da altura da imagem, respectivamente. Muitos diagramadores utilizam esses atributos para realizar um redimensionamento da imagem, aumentando-a (com resultados precários, deixam a imagem “pixelada”) ou diminuindo-a, o que é ainda mais grave. Usar uma imagem grande demais e redimensioná-la no HTML não torna a imagem de origem menor; o arquivo pesado terá que ser baixado pelo usuário, causando problemas de performance. Aconselha-se criar arquivos diferentes da mesma imagem em dimensões diferentes quantas vezes forem necessárias para uso.

Para que servem **width e height**, então? Diagramação. Se, por acaso, a imagem de origem não for localizada, esses atributos garantem que a largura e a altura sejam reservadas para o local que a imagem ocuparia, evitando que haja uma quebra de diagramação do documento. Os atributos devem, portanto, ser informados sempre com a largura e a altura reais da imagem.

**alt:** texto alternativo da imagem, importantíssimo para a acessibilidade de um documento. É apresentado se a imagem de origem não for localizada; o texto assume o lugar da imagem. É este mesmo texto que será utilizado pelo leitor de tela de um deficiente visual, que poderá receber uma descrição da imagem que não pode ver.

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de imagem</title>
    <meta charset="utf-8">
  </head>
  <body>
    
  </body>
</html>
```

Código-fonte 21 – Tag <img> aplicada ao HTML  
Fonte: Elaborado pelo autor (2016)



Figura 7 – Tag <img> em dois momentos: com a imagem origem não localizada e com a imagem localizada

Fonte: Elaborado pelo autor (2016)

### 1.6.5 Tag <div>

A tag <div> representa uma divisão ou seção do documento HTML. Trata-se de um contêiner utilizado para agrupar elementos, como texto e imagens. Com o CSS, esse contêiner pode ser posicionado em qualquer parte da página (esteticamente falando), tornando-se uma verdadeira revolução na diagramação de páginas.


Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de div</title>
    <meta charset="utf-8">
  </head>
  <body>
    <div class="artigo">Conteúdo Teste representando um
artigo.</div>
  </body>
</html>
```

Código-fonte 22 – Tag <div> aplicada ao HTML

Fonte: Elaborado pelo autor (2016)





Conteúdo teste representando um artigo.

Figura 8 – Tag <div> em funcionamento  
Fonte: Elaborado pelo autor (2016)

Abordaremos com detalhes as questões de posicionamento e estilização do texto no capítulo de CSS.

#### 1.6.6 SUBSTITUTIVOS DO <DIV> PARA MELHORAR A SEMÂNTICA

Muitos autores encorajam fortemente o uso das novas *tags* que foram criadas no HTML 5 (como <article>, <section>, entre outras), reservando a *tag* <div> apenas aos casos em que essas novas *tags* não parecem ser aplicáveis. Todas elas se comportam como a *tag* <div>, mas facilitam a manutenção dos documentos, além de melhorarem a compreensão do texto por parte dos mecanismos de busca. São elas:

**<section>:** representa a seção de um documento ou página, podem ser capítulos ou seções de uma tese.

**<nav>:** um contêiner de *links* para navegação, como um menu.

**<article>:** contém o artigo propriamente dito, trata-se do conteúdo do documento, podendo ser um artigo de jornal ou revista, uma postagem de fórum, entre outros.

**<aside>:** este contêiner ficará ao lado, tangenciando o conteúdo principal (geralmente um artigo). Pode ser usado para a publicidade ou um conteúdo de menor importância, pois o <aside> é um dos primeiros elementos a sumirem em dimensões de tela menores quando nos referimos a um *layout* responsivo.

**<hgroup>:** é o título da seção. Por ser um agrupamento, pode abranger as *tags* <h1> até <h6>, contendo, assim, subtítulos, por exemplo.

**<header>**: contêiner de cabeçalho do documento (ou de um artigo), pode possuir menu de navegação (<nav>), o título (com elemento <hgroup> e/ou <h1>-<h6>), entre outros.

**<footer>**: geralmente utilizados no fim de uma seção, representam o rodapé de uma seção ou do documento como um todo. Podem conter *links* relacionados, referências de direitos autorais, entre outros.

**<time>**: representa não só a hora, mas uma data do calendário gregoriano; é muito útil aos mecanismos de busca, pois indica a idade da informação ali contida (considerando que será usado em um contexto, como um artigo).

**<mark>**: serve para dar ênfase ou destaque em uma parte do texto ou documento, como referência.

**<figure>**: indica a presença de uma figura no contêiner, podendo conter uma legenda.

**<figcaption>**: legenda de uma figura, deve ser usado dentro da *tag* <figure>.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de Título</title>
    <meta charset="utf-8">
  </head>
  <body>
    <article>Conteúdo Teste representando um
    artigo.</article>
  </body>
</html>
```

Código-fonte 23 – Exemplo anterior trocando <div> por <article>  
Fonte: Elaborado pelo autor (2016)

```
<!DOCTYPE html>
<html lang="pt-br">
  <header>
    <meta charset="utf-8">
  </header>
  <body>
    <article>
      <header>
        <h1>Título do Artigo</h1>
        <p>Publicado em: <time pubdate="pubdate"
        datetime="2015-11-17">17 de novembro de 2015</time></p>
      </header>
```

```
<p>Aqui começa este grande artigo...</p>
<section>
  <h2>Subseção do Artigo</h2>
  <p>Aqui entra uma subseção...</p>
</section>
<section>
  <h2>Outra subseção do Artigo</h2>
  <p>Aqui entra outra subseção...</p>
  <figure>
    
    <figcaption>Figura importante da
seção</figcaption>
  </figure>
</section>
<footer>
  <p>Creative Commons Attribution-
ShareAlike License</p>
</footer>
</article>
</body>
</html>
```

Código-fonte 24 – Exemplo de uso das novas *tags* semânticas  
Fonte: Elaborado pelo autor (2016)

### 1.6.7 TAG <A>

Utilizada para fazer uma âncora de *hyperlink*, ou seja, o *hyperlink* propriamente dito, tão tradicional na Internet. A seguir, seus principais atributos:

**href:** *hyperlink reference*, ou seja, o endereço do documento em que o usuário seja submetido caso clique no *link*. Pode ser um endereço relativo, apontando para o mesmo servidor de hospedagem, ou absoluto, indicando para *links* externos (outros domínios). Nos primórdios, a *tag* era utilizada para fazer um *link* dentro do mesmo documento, para um ponto mais adiante (ou mesmo para trás), usando âncoras. Essa abordagem caiu em desuso, sendo preferível, por várias razões, quebrar um documento HTML muito grande em vários documentos menores (embora seja mais trabalhoso).

**download:** a presença deste atributo na *tag* <a> indica que o autor deseja que o usuário baixe o documento (seja ele HTML ou não) em vez de ser submetido a ele.

**rel:** utilizado para qualificar o relacionamento entre os documentos vinculados. Vide material específico de HTML 5 para detalhes.

**target:** define o alvo do *hyperlink*. O valor-padrão é “\_self”, ou seja, ao clicar no *hyperlink* a página ou documento informado em *href* será carregada na mesma janela em que o documento atual está. Outras possibilidades são: “\_blank”, cujo destino é aberto em uma nova janela ou aba; “\_parent”, documento abre no frame pai (quando há uso de frames); e “\_top”, documento de destino ignora a divisão de frames da tela e abre em toda a área de janela disponível. Sendo assim, finalmente, podemos informar o nome do frame, no qual desejamos que a página ou documento destino seja aberto.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de hyperlink</title>
    <meta charset="utf-8">
  </head>
  <body>
    <a href="outra página">Link para outra página</a>
  </body>
</html>
```

Código-fonte 25 – Tag <a> implementada em HTML  
Fonte: Elaborado pelo autor (2016)

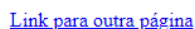


Figura 9 – Tag <a> em funcionamento  
Fonte: Elaborado pelo autor (2016)

Imagens podem ser utilizadas como *hyperlink*, combinando as tags <a> e <img>:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de hyperlink</title>
    <meta charset="utf-8">
  </head>
  <body>
    <a href="http://www.fiap.com.br/">
    </a>
  </body>
</html>
```

Código-fonte 26 – Tags <a> e <img> criando um *hyperlink* em uma imagem  
Fonte: Elaborado pelo autor (2016)



Figura 10 – Hyperlink aplicado em uma imagem  
Fonte: Elaborado pelo autor (2016)

### 1.6.8 TAGS <OL>, <UL> E <LI>

As tags <ol> e <ul> são utilizadas para a criação de listas ordenadas e não ordenadas, respectivamente. A tag <li> é usada para os itens da lista, independentemente do formato.

Exemplo de lista ordenada:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de ol</title>
    <meta charset="utf-8">
  </head>
  <body>
    <ol>
      <li value="1">Teste 1</li>
```

```
<li value="2">Teste 2</li>
<li value="3">Teste 3</li>
</ol>
</body>
</html>
```

Código-fonte 27 – Tags <ol> e <li> criando uma lista ordenada  
Fonte: Elaborado pelo autor (2016)



Figura 11 – Lista ordenada  
Fonte: Elaborado pelo autor (2016)

Exemplo de lista não ordenada:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de ul</title>
    <meta charset="utf-8">
  </head>
  <body>
    <ul>
      <li>Teste 1</li>
      <li>Teste 2</li>
      <li>Teste 3</li>
    </ul>
  </body>
</html>
```

Código-fonte 28 – Tags <ul> e <li> criando uma lista não ordenada  
Fonte: Elaborado pelo autor (2016)



Figura 12 – Lista não ordenada  
Fonte: Elaborado pelo autor (2016)

### 1.6.9 TAG <IFRAME>

Permite criar uma janela dentro do documento HTML na qual é possível abrir outro documento HTML. Os principais atributos são a largura da janela (*width*), sua altura (*height*), o nome da janela (*name*) e o endereço de Internet deste outro documento (*src*).

Veja o exemplo a seguir. Ele precisa de três documentos HTML para funcionar: `iframe_pagina1.html`, `iframe_pagina2.html` e `iframe.html`. Este último possui um *iframe* que carregará previamente a primeira página e tem também um *link* com *target*, ao clicar, a página 2 será aberta dentro do *iframe*:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de iframe</title>
    <meta charset="utf-8">
  </head>
  <body>Página 1</body>
</html>
```

Código-fonte 29 – O arquivo `iframe_pagina1.html`  
Fonte: Elaborado pelo autor (2016)

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de iframe</title>
    <meta charset="utf-8">
  </head>
  <body>Página 2</body>
</html>
```

Código-fonte 30 – O arquivo iframe\_pagina2.html

Fonte: Elaborado pelo autor (2016)

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de iframe</title>
    <meta charset="utf-8">
  </head>
  <body>
    <a href="iframe_pagina2.html" target="janela">Clique
aqui para abrir a página 2 dentro do iframe</a>
    <br><br>
    Iframe abaixo:
    <iframe name="janela" width="100" height="100"
src="iframe_página1.html">
    </iframe>
  </body>
</html>
```

Código-fonte 31 – O arquivo iframe.html

Fonte: Elaborado pelo autor (2016)



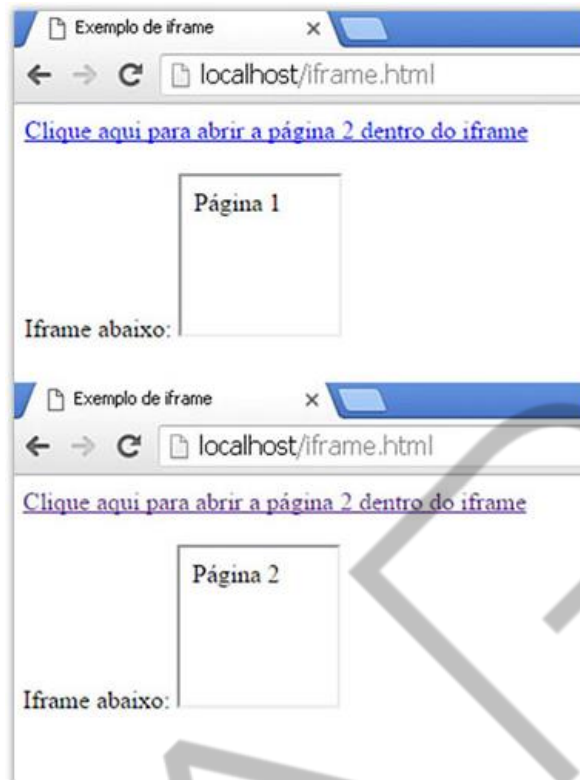


Figura 13 – *iframe* em dois momentos: na carga da página e ao clicar no *hyperlink* disponível  
Fonte: Elaborado pelo autor (2016)

#### 1.6.10 Tags `<frame>` e `<frameset>`: por que não usar?

As *tags* em questão permitem dividir uma janela de navegador em vários pedaços, possibilitando a abertura de vários documentos HTML distintos ao mesmo tempo. Usados à exaustão nas décadas de 1990 e início da década seguinte, pesam demais na performance e são de difícil controle: cada janela é batizada com um nome e os *hyperlinks* com a *tag* `<a>` devem usar o atributo *target* para abrir conteúdos nas outras janelas.

Em uma Internet acessada em múltiplos dispositivos e com uma preocupação em ser responsiva (abrir o conteúdo adequadamente em diferentes tamanhos de tela), usar *frames* é um retrocesso.

Se o desenvolvedor quiser reaproveitar todo o *layout* de um documento HTML, mudando apenas parte dele de tempos em tempos (e sem a necessidade de

renderizar páginas o tempo todo), é aconselhável que seja feito utilizando JavaScript e Ajax, e isso será abordado no conteúdo devido.

### 1.6.11 TAGS DE MARCAÇÃO A SE ESQUECER

Nos primórdios, existiam *tags* importantes para a marcação de texto, como as *tags* `<font>` (formatação de fontes de texto); `<big>` (fonte maior); `<small>` (fonte menor); `<b>` (negrito); `<i>` (itálico); `<s>` (tachado); e `<u>` (sublinhado). Com exceção de `<b>`, `<i>` e `<u>`, todas as *tags* mencionadas foram descontinuadas no HTML 5. A razão é que a melhor forma de se estilizar texto se chama CSS: por ser mais organizado e possibilitar o reúso dos estilos. Esqueça que essas *tags* existem (ou existiram): utilize a *tag* `<div>` ou uma *tag* conhecida como `<span>` com o atributo *class* e deixe tudo a cargo do CSS, como você aprenderá a fazer no capítulo devido.

## 2 TABULAÇÃO DE DADOS: CRIANDO TABELAS

Desde a origem das teses em hipertexto, uma necessidade se mantém constante: a de tabular dados para apresentação. Esta é a única boa razão para o uso de tabelas. Digo isso porque antes da popularização da *tag* `<div>` e da evolução do CSS, tabelas eram utilizadas para diagramar o *layout* da página, por permitirem personalizar suas dimensões e colocar tabelas umas dentro de outras. A esta reformulação (deixar de usar tabelas e passar a utilizar *div's* posicionados com CSS), damos o nome de *tableless*.

Não quer dizer, no entanto, que tabelas são más e devem ser enterradas a sete palmos. Elas devem ser usadas para o que se propõem: tabular dados!

### 2.1 Tag `<table>`

Utilizar para determinar o início e o fim (`</table>`) de uma tabela. Embora possamos usar os atributos *width* e *height* para dimensionar a tabela – seja em seu tamanho exato em pixels ou percentualmente –, alinhá-la usando *align* e determinar a espessura de sua borda com *border*, tudo isso é feito com melhores resultados usando CSS. Os exemplos a seguir utilizarão os atributos, mas procure fazê-las usando os estilos que o CSS lhe oferece.

Eis o primeiro exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8">
  </head>
  <body>
    <table width="100%" height="200">
    </table>
  </body>
</html>
```

Código-fonte 32 – Tag `<table>` sendo utilizada  
Fonte: Elaborado pelo autor (2016)

E temos o seguinte resultado:

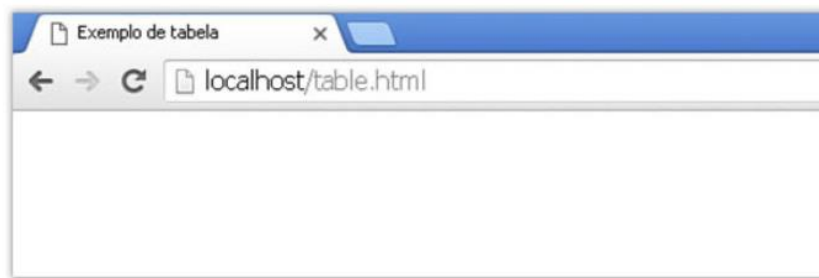


Figura 14 – Tabela criada  
Fonte: Elaborado pelo autor (2016)

Exatamente, nada! Sei que é frustrante, mas não temos muito que ver por enquanto... precisamos de outras *tags* para termos algum resultado.

## 2.2 Tags <thead>, <tbody> e <tfoot>

Embora sejam *tags* consideradas opcionais, ajudam a definir bem as áreas distintas de uma tabela.

<thead> indica o início do cabeçalho da tabela, <tbody> seu corpo e <tfoot> seu rodapé. Usá-las possibilita, por exemplo, determinar que as duas primeiras linhas da tabela são seu cabeçalho, e não apenas a primeira.

Veja o exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8">
  </head>
  <body>
    <table width="100%" height="200">
      <thead>
      </thead>
      <tbody>
      </tbody>
      <tfoot>
      </tfoot>
    </table>
  </body>
</html>
```

Código-fonte 33 – Tags <thead>, <tbody> e <tfoot> sendo utilizadas  
Fonte: Elaborado pelo autor (2016)

Segundo a especificação, as *tags* de finalização são consideradas opcionais. Sendo assim, o navegador compreende que o cabeçalho <thead> termina assim que ele encontra um <tbody>; e o corpo da tabela acaba quando se depara com o <tfoot>, que se encerra com o final da tabela:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8">
  </head>
  <body>
    <table width="100%" height="200">
      <thead>
      <tbody>
      <tfoot>
    </table>
  </body>
</html>
```

Código-fonte 34 – Tags <thead>, <tbody> e <tfoot> sem suas tags de finalização  
Fonte: Elaborado pelo autor (2016)

Embora codificar assim sempre me dá a impressão de que está faltando alguma coisa...

## 2.3 Tag <tr>

Utilizada para determinar o início e o fim (</tr>) de uma linha da tabela, que poderá ter inúmeras linhas:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <table width="100%" height="200">
      <thead>
        <!-- Primeira linha -->
        <tr></tr>
        <!-- Segunda linha -->
        <tr></tr>
      </thead>
      <tbody>
```

```
        <!-- Terceira linha -->
        <tr></tr>
        <!-- Quarta linha -->
        <tr></tr>
    </tbody>
    <tfoot>
        <!-- Quinta linha -->
        <tr></tr>
    </tfoot>
</table>
</body>
</html>
```

Código-fonte 35 – Linhas de tabela sendo definidas com a tag <tr>  
Fonte: Elaborado pelo autor (2016)

Neste exemplo, temos duas no cabeçalho, duas no corpo da tabela e uma no rodapé, e nada ainda para ver. Tenha paciência!

## 2.4 Tags <td> e <th>

Ambas são utilizadas para determinar o início e o fim de uma célula de tabela. Mas você se pergunta: “E as colunas?” Bem, uma tabela HTML possui tantas colunas quanto sua linha que possui o maior número de tags <td>. Sendo assim, se em uma linha de tabela temos 3 <td>s e na segunda 4 <td>s, a tabela possui 4 colunas, e teremos uma lacuna na primeira.

Tags <td> e <th> são similares: <td> deve ser usada em células comuns, enquanto <th> deve ser utilizada para células de cabeçalho. A princípio, não há diferença visual entre elas (embora alguns navegadores deixem a fonte dentro de um <th> em negrito), mas você pode fazê-las com CSS, veremos isso. Entretanto, o uso do <th> traz outros benefícios, como repetir o cabeçalho automaticamente ao se mandar imprimir uma grande tabela.

Eis o primeiro exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <table width="500" border="1">
```

```

<thead>
  <tr>
    <!-- Primeira linha -->
    <th>Curso</th>
    <th>Conteudista</th>
    <th>Tutor</th>
    <th>Realizado</th>
  </tr>
</thead>
<tbody>
  <tr>
    <!-- Segunda linha -->
    <td>Gamificação</td>
    <td>Henrique Poyatos</td>
    <td>Henrique Poyatos</td>
    <td>Sim</td>
  </tr>
  <tr>
    <!-- Terceira linha -->
    <td>Prototipação</td>
    <td>Almir Alves</td>
    <td>Almir Alves</td>
    <td>Sim</td>
  </tr>
</tbody>
</table>
</body>
</html>

```

Código-fonte 36 – Exemplo de tabelas usando <td> e <th>  
 Fonte: Elaborado pelo autor (2016)

Ufa! Finalmente consigo lhe mostrar algo:

Curso	Conteudista	Tutor	Realizado
Gamificação	Henrique Poyatos	Henrique Poyatos	Sim
Prototipação	Almir Alves	Almir Alves	Sim

Figura 15 – Tabela completa  
 Fonte: Elaborado pelo autor (2016)

## 2.5 Dimensionando células

É possível mudar a dimensão das células (e, por consequência, das colunas). Se quisermos que a primeira coluna possua 50% do tamanho total da tabela, basta colocar **width="50%"** em um dos <td>s ou <th>s que se apresentam logo após os <tr>s (primeira célula da linha, portanto):

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <table width="500" border="1">
      <thead>
        <tr>
          <!-- Primeira linha -->
          <th width="50%">Curso</th>
          <th>Conteudista</th>
          <th>Tutor</th>
          <th>Realizado</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <!-- Segunda linha -->
          <td>Gamificação</td>
          <td>Henrique Poyatos</td>
          <td>Henrique Poyatos</td>
          <td>Sim</td>
        </tr>
        <tr>
          <!-- Terceira linha -->
          <td>Prototipação</td>
          <td>Almir Alves</td>
          <td>Almir Alves</td>
          <td>Sim</td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```

Código-fonte 37 – Mudando as dimensões de uma célula (e coluna)

Fonte: Elaborado pelo autor (2016)



Curso	Conteudista	Tutor	Realizado
Gamificação	Henrique Poyatos	Henrique Poyatos	Sim
Prototipação	Almir Alves	Almir Alves	Sim

Figura 16 – Dimensionando a primeira célula (e coluna) em 50% do tamanho total da tabela  
Fonte: Elaborado pelo autor (2016)

Existe um certo limite, no entanto. Se colocássemos a primeira célula (e coluna) em 70% ou 80%, as palavras “Conteudista” e “Realizado” não permitiriam “espremer” a segunda e a quarta colunas ainda mais; o navegador faria a primeira no máximo percentual possível.

## 2.6 Mesclando colunas

As *tags* `<td>` e `<th>` possuem atributos que permitem mesclá-las com outras: são o ***colspan*** (para mesclar colunas horizontalmente) e ***rowspan*** (para mesclar linhas na vertical).

E se quisermos, por exemplo, mesclar as colunas de Conteudista e Tutor nas células que se repetem, como “Henrique Poyatos”? Para isso, precisamos colocar um **`colspan="2"`** no segundo `<td>` da segunda linha. Além disso, precisamos **eliminar** o terceiro `<td>` da segunda linha. Sim, isso mesmo: o segundo `<td>` agora vale por dois:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de tabela</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <table width="500" border="1">
      <thead>
        <tr>
```

```

        <!-- Primeira linha -->
        <th width="50%">Curso</th>
        <th>Conteudista</th>
        <th>Tutor</th>
        <th>Realizado</th>
    </tr>
</thead>
<tbody>
    <tr>
        <!-- Segunda linha -->
        <td>Gamificação</td>
        <td colspan="2">Henrique
Poyatos</td>
        <td>Sim</td>
    </tr>
    <tr>
        <!-- Terceira linha -->
        <td>Prototipação</td>
        <td>Almir Alves</td>
        <td>Almir Alves</td>
        <td>Sim</td>
    </tr>
</tbody>
</table>
</body>
</html>

```

Código-fonte 38 – Mesclando células horizontalmente.  
Fonte: Elaborado pelo autor (2021)

Curso	Conteudista	Tutor	Realizado
Gamificação	Henrique Poyatos		Sim
Prototipação	Almir Alves	Almir Alves	Sim

Figura 17 – Mesclando células horizontalmente  
Fonte: Elaborado pelo autor (2021)

Algo similar é feito ao mesclar as duas colunas com a palavra “Sim”. Desta vez, usaremos *rowspan*:

```

<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <title>Exemplo de tabela</title>
        <meta charset="utf-8" />
    </head>
    <body>
        <table width="500" border="1">
            <thead>

```

```

<tr>
  <!-- Primeira linha -->
  <th width="50%">Curso</th>
  <th>Conteudista</th>
  <th>Tutor</th>
  <th>Realizado</th>
</tr>
</thead>
<tbody>
  <tr>
    <!-- Segunda linha -->
    <td>Gamificação</td>
    <td colspan="2">Henrique
Poyatos</td>
    <td rowspan="2">Sim</td>
  </tr>
  <tr>
    <!-- Terceira linha -->
    <td>Prototipação</td>
    <td>Almir Alves</td>
    <td>Almir Alves</td>
  </tr>
</tbody>
</table>
</body>
</html>

```

Código-fonte 39 – Mesclando células verticalmente  
Fonte: Elaborado pelo autor (2016)

Curso	Conteudista	Tutor	Realizado
Gamificação	Henrique Poyatos		Sim
Prototipação	Almir Alves	Almir Alves	

Figura 18 – Mesclando células verticalmente  
Fonte: Elaborado pelo autor (2021)

## 3 CRIANDO FORMULÁRIOS

A principal forma de interação, no que diz respeito à entrada de dados, são os formulários. Desde as primeiras versões do HTML, são a forma mais eficaz (quando não a única) de solicitar informações do usuário.

### 3.1 Tag <form>

Utilizada para determinar o início e o fim de um formulário HTML. Todas as *tags* seguintes estarão contidas no formulário, ou seja, entre <form> e </form>. Além dos atributos globais, os listados a seguir estão disponíveis para parametrização:

**accept-charset:** usada para armazenar o padrão de caracteres que será utilizado ao submeter o formulário; procure usar utf-8 e, em último caso, iso-8859-1.

**action:** utilizada para armazenar o endereço de Internet (URL) no qual o formulário será submetido. Embora aceite caminhos absolutos (além dos relativos), o ideal é que essa URL de tratamento do formulário esteja em um mesmo domínio, caso contrário, teremos uma falha de segurança conhecida como **cross-origin**.

**autocomplete:** permite que as funcionalidades de autocompletar presentes no navegador auxiliem o usuário no preenchimento do formulário.

**enctype:** define qual tipo de codificação será utilizado para submeter o formulário. O padrão é “application/x-www-form-urlencoded”, opção a partir da qual os dados são preparados para fazer parte da URL (método GET de envio). As outras possibilidades são “multipart/form-data”, necessária quando requerer a realização de *upload* de arquivos usando o formulário (os dados se tornam binários), e “text/plain”, na qual espaços se tornam sinais de adição (“+”), mas nenhum outro tratamento é aplicado.

**method:** define qual método HTTP será usado para submeter o formulário. Existem duas opções: GET e POST.

O método GET, embora seja o padrão, não é a situação ideal. Ao utilizá-lo, ele transfere as informações do formulário para o endereço URL seguinte. Isso implica

limitações no tamanho da informação passada, além de possíveis problemas de segurança.

No método POST, as informações são enviadas do corpo de mensagem do protocolo HTTP, possibilitando volumes maiores de informação e ocultando (um pouco) a informação dos usuários.

**Nota:** a não ser que o protocolo usado seja HTTPS (estabelece um túnel criptografado entre cliente e servidor), seja GET ou POST, as informações são enviadas de forma aberta, podendo ser interceptadas por qualquer nó de rede, na qual os pacotes HTTP são transmitidos. Conclusão, métodos POST são menos inseguros do que métodos GET.

**name:** define o nome que será dado ao formulário dentro da API `document.forms` (JavaScript). Útil, mas dê preferência para o atributo `id`, que é mais prático.

**novalidate:** se este atributo estiver presente em `<form>` (não precisa de valor definido), a validação do formulário (que acontece antes da submissão) é ignorada.

**target:** o mesmo atributo apresentado na tag `<a>`.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST"
action="processaForm.php">
    </form>
  </body>
</html>
```

Código-fonte 40 – Exemplo de aplicação da tag `<form>` em HTML  
Fonte: Elaborado pelo autor (2016)

A exemplo do que aconteceu com as tabelas, ainda não há muito o que se ver.

### 3.2 Tag <label>

Trata-se do rótulo do campo, uma descrição do que aquele elemento de formulário solicita ou representa. Ele se torna eficaz quando utilizado com o atributo *form*: este permite associar o <label> com seu elemento de formulário (seja <input>, <select>, <textarea> e outros), bastando informar o *id* do elemento.

Exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST"
action="processaForm.php">
      <label for="nome">Nome Completo</label>
      <input type="text" id="nome"><!-- explicado na
sequência -->
    </form>
  </body>
</html>
```

Código-fonte 41 – Exemplo de uso da tag <label> em HTML  
Fonte: Elaborado pelo autor (2016)



Figura 19 – Rótulo para campos de formulário  
Fonte: Elaborado pelo autor (2016)

Ao clicar no <label> “Nome completo” com o mouse, a caixa de texto ganha foco imediatamente.

### 3.3 Tag <input>

É o principal elemento de um formulário e pode assumir a forma de uma caixa de texto ou de um botão de seleção única ou múltipla, de um botão de formulário, entre outras possibilidades. Seus principais atributos são:

**accept:** funciona como dica de quais tipos de arquivo serão aceitos em um elemento de *upload* (`type="file"`).

**alt:** o mesmo apresentado em <img>, utilizado no caso de `type="image"`.

**autocomplete:** habilita a funcionalidade de autopreenchimento.

**autofocus:** concede o foco ao elemento automaticamente ao carregar a página; apenas um dos elementos de formulário pode possuir este atributo.

**checked:** ao colocar este atributo em um elemento que seja `type="checkbox"` ou `type="radio"`, o elemento vem marcado como padrão; do contrário, ele vem desmarcado.

**disabled:** ao colocar este atributo no elemento, indica que ele está indisponível no momento e fica indisponível para alterações.

**height:** utilizado para definir a altura do elemento. Prefira seu equivalente em CSS.

**list:** informa uma lista de opções para o recurso de autocompletar (utiliza a tag <datafield>, a ser estudada adiante).

**max:** novo atributo, define o valor máximo que um campo do tipo numérico ou data (formato yyyy-mm-dd) pode assumir.

**maxlength:** tamanho máximo, em caracteres, esperado para o valor preenchido.

**min:** novo atributo, define o valor mínimo que um campo do tipo numérico ou data (formato yyyy-mm-dd) pode assumir.

**minlength:** tamanho mínimo, em caracteres, esperado para o valor preenchido.

**multiple:** permite ao campo coletar vários valores de uma vez. Campo múltiplo é utilizado pelo navegador quando o `type="file"` ou `type="email"`.

***name***: define o nome do elemento na API `form.elements` (JavaScript). Este elemento é particularmente importante em uma linguagem como PHP, cujas APIs de tratamento também usam nome. Defina também *id*, pois é mais prático ao usar JavaScript.

***pattern***: novo atributo, permite definir um padrão em expressões regulares para validar este campo de formulário.

***placeholder***: trata-se de um rótulo visível para o usuário, posicionado dentro do elemento. Geralmente, é utilizado como dica de preenchimento.

***readonly***: ao colocar este atributo no elemento, indica que ele é apenas leitura e fica indisponível para alterações.

***required***: novo atributo, ao informá-lo indica que o campo é de preenchimento obrigatório.

***size***: tamanho do campo em número de caracteres. Prefira usar *width* em CSS.

***type***: define qual é o tipo de elemento de formulário. As opções são:

- ***text***: caixa de texto padrão.
- ***hidden***: campo “escondido” ou “invisível”. Não é renderizado.
- ***password***: caixa de texto que mascara os caracteres digitados.
- ***checkbox***: uma caixa para seleção múltipla.
- ***radio***: um botão redondo para seleção única.
- ***file***: uma caixa de texto e botão que permitem procurar arquivos no repositório local do usuário. Campo para *upload* de arquivos.
- ***submit***: um botão que, ao ser clicado, submete o formulário.
- ***reset***: um botão que, ao ser clicado, restaura o formulário para os seus valores iniciais, ou seja, geralmente apaga o formulário.
- ***button***: um botão que não faz nada ao ser clicado; é específico para ser trabalhado usando JavaScript.



Existem vários tipos novos criados no HTML 5:

**tel:** específico para armazenar telefone.

**url:** específico para armazenar um endereço web (URL).

**search:** específico para campo de pesquisa.

**email:** específico para armazenar um endereço de e-mail, valida automaticamente.

**number:** específico para armazenar valores numéricos.

**date:** específico para armazenar data.

**time:** específico para armazenar hora.

**range:** específico para intervalo numérico.

**color:** específico para armazenar cores.

**value:** permite definir um valor-padrão para o campo. Dessa maneira, ao renderizar o formulário, o campo vem previamente preenchido.

**width:** utilizado para definir a largura do elemento. Prefira seu equivalente em CSS.

O exemplo abaixo utiliza os atributos *value* e *disabled*:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <label for="nome">Nome Completo: </label>
      <input type="text" id="nome" value="João Ninguém"
disabled>
    </form>
  </body>
</html>
```

Código-fonte 42 – Exemplo de formulário usando <input> com *value* e *disabled*

Fonte: Elaborado pelo autor (2016)

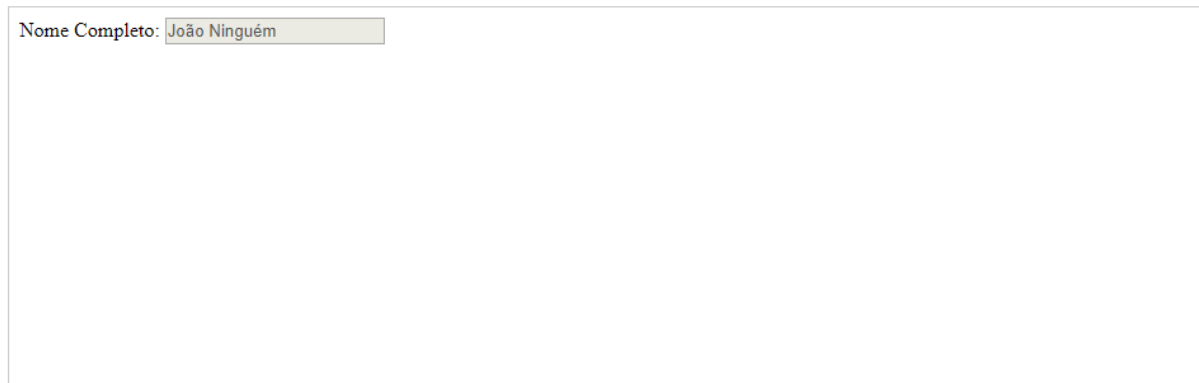
A screenshot of a web form. At the top, there is a label "Nome Completo:" followed by a text input field containing the text "João Ninguém". The input field has a light gray border and a subtle shadow. Below the input field, there is a large, empty rectangular area, likely a placeholder for a larger form element or a message.

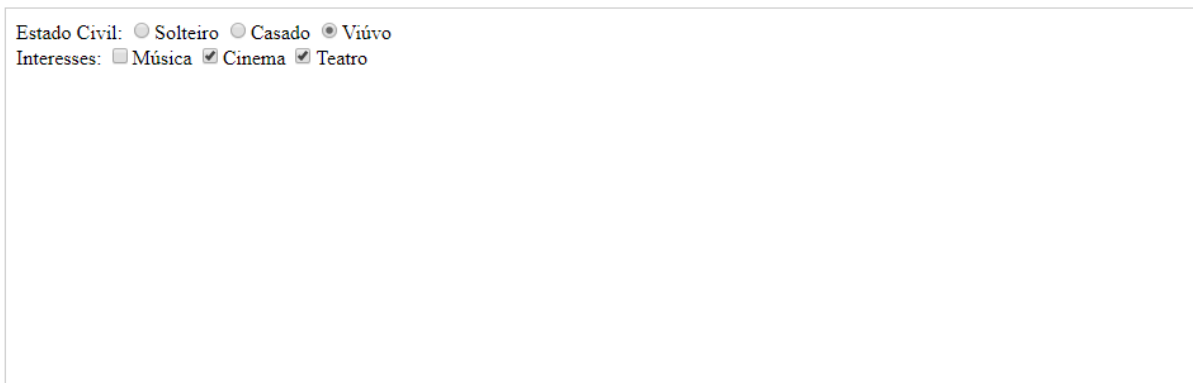
Figura 20 – Exemplo de formulário usando `<input>` com *value* e *disabled*  
Fonte: Elaborado pelo autor (2016)

No exemplo a seguir, botões de seleção única e caixas de múltipla seleção usando o atributo *checked*:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <!-- Botões de escolha única -->
      <label for="estcivil">Estado Civil: </label>
      <input type="radio" name="estcivil"
value="Solteiro">Solteiro
      <input type="radio" name="estcivil"
value="Casado">Casado
      <input type="radio" name="estcivil" value="Viúvo"
checked>Viúvo<br>

      <!-- Caixas de múltipla escolha -->
      <label for="interesses">Interesses: </label>
      <input type="checkbox" name="interesses"
value="Música">Música
      <input type="checkbox" name="interesses"
value="Cinema" checked>Cinema
      <input type="checkbox" name="interesses"
value="Teatro" checked>Teatro<br>
    </form>
  </body>
</html>
```

Código-fonte 43 – Exemplo de formulário usando `<input>` com *checked*  
Fonte: Elaborado pelo autor (2016)



Estado Civil: ☐ Solteiro ☐ Casado ☒ Viúvo  
Interesses: ☐ Música ☒ Cinema ☒ Teatro

Figura 21 – Exemplo de formulário usando `<input>` com *checked*  
Fonte: Elaborado pelo autor (2016)

Repare que os botões de escolha única (`type="radio"`) possuem os mesmos atributos *name*; é dessa maneira que esses botões são agrupados, permitindo que apenas um deles seja marcado. No caso de caixas de múltipla escolha (`type="checkbox"`), isso é desejável, embora não seja obrigatório.

Veja um exemplo utilizando todos os tipos de `<input>` mais tradicionais:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <!-- Caixa de texto -->
      <label for="nome">Nome Completo: </label>
      <input type="text" id="nome"><br>
      <!-- Caixa para senha -->
      <label for="senha">Senha: </label>
      <input type="password" id="senha"><br>
      <!-- Botões de escolha única -->
      <label for="estcivil">Estado Civil: </label>
      <input type="radio" name="estcivil"
value="Solteiro">Solteiro
      <input type="radio" name="estcivil"
value="Casado">Casado
      <input type="radio" name="estcivil"
value="Viúvo">Viúvo<br>
      <!-- Caixas de múltipla escolha -->
      <label for="interesses">Interesses: </label>
      <input type="checkbox" name="interesses"
value="Música">Música
```

```

        <input type="checkbox" name="interesses"
value="Cinema">Cinema
        <input type="checkbox" name="interesses"
value="Teatro">Teatro<br>
        <!-- Campo para upload -->
        <label for="foto">Foto: </label>
        <input type="file" id="foto"><br>
        <!-- Botões -->
        <input type="submit" value="Enviar">
        <input type="reset" value="Limpar dados">
        <input type="button" value="Faz nada">
    </form>
</body>
</html>

```

Código-fonte 44 – Exemplo de formulário usando <input>  
Fonte: Elaborado pelo autor (2016)

Figura 22 – Exemplo de formulário usando <input>  
Fonte: Elaborado pelo autor (2016)

Veja o material específico de HTML 5 para outros exemplos abordando os novos types e atributos.

### 3.4 Tags <select>, <option> e <optgroup>

Essas três *tags* são utilizadas para criar caixas de seleção (conhecidas como “ComboBox”). A *tag* <select> define o início e o fim da lista, que pode ser parametrizada com os seguintes atributos:

atributos globais, *autofocus*, *disabled*, *name* e *required*: funcionamento idêntico ao explicado na *tag* <input>.

**multiple:** permite escolher mais de um valor na caixa de seleção.

**size:** tamanho da lista em número de linhas, transformando-a em uma lista.

No entanto, `<select>` precisa conter *tags* `<option>` para que faça sentido. Elas correspondem às opções que o usuário terá para escolher nessa caixa de seleção. Os atributos fundamentais são:

**disabled:** desabilita a opção, tornando-a indisponível.

**value:** define o valor da opção, ou seja, que valor será submetido caso aquela opção seja selecionada.

**selected:** permite definir uma ou mais opções (caso `<select>` seja *multiple*) que começarão previamente selecionadas.

Veja um exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <select id="estcivil" name="estcivil">
        <option value="S">Solteiro</option>
        <option value="C">Casado</option>
        <option value="V" selected>Viúvo</option>
        <option value="D">Divorciado</option>
      </select>
    </form>
  </body>
</html>
```

Código-fonte 45 – Exemplo de formulário usando `<select>` e `<option>` com opção previamente selecionada

Fonte: Elaborado pelo autor (2016)

Figura 23 – Exemplo de formulário usando <select> e <option> com opção previamente selecionada  
Fonte: Elaborado pelo autor (2016)

Caso esse formulário fosse submetido, apenas as iniciais dos estados civis – “S”, “C”, “V”, e “D” – seriam enviadas adiante, bem diferente do rótulo visível ao usuário contido dentro da *tag* <option>.

Conforme já mencionado, podemos criar listas para múltipla seleção: basta utilizar os atributos *multiple* e *size*. Aproveitemos o exemplo para mostrar o funcionamento da *tag* <optgroup>, que agrupa as opções em categorias:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <label for="carros">Carros preferidos</label>
      <select id="carros" name="carros" size="8"
multiple>
        <optgroup label="FIAT">
          <option value="500">500</option>
          <option value="Uno">Palio</option>
          <option value="Uno">Uno</option>
        </optgroup>
        <optgroup label="FORD">
          <option value="Fiesta">Fiesta</option>
          <option value="Focus">Focus</option>
          <option value="Ka">Ka</option>
        </optgroup>
      </select>
    </form>
  </body>
</html>
```

Código-fonte 46 – Exemplo de formulário usando <select>, <option> e <optgroup> com seleção múltipla

Fonte: Elaborado pelo autor (2016)

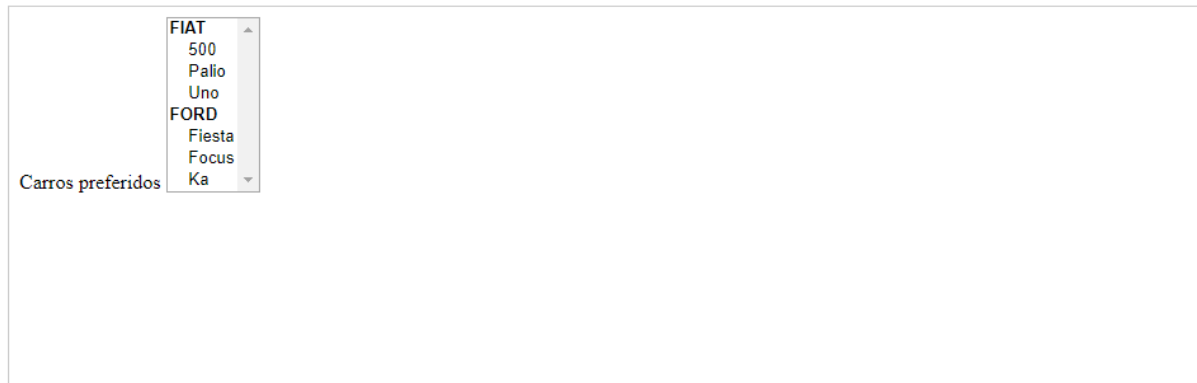


Figura 24 – Exemplo de formulário usando `<select>`, `<option>` e `<optgroup>` com seleção múltipla  
 Fonte: Elaborado pelo autor (2016)

### 3.5 Tag `<textarea>`

Utilizada quando é necessário que o usuário escreva um longo texto, grande demais para caixas de texto comuns. Na sequência, os principais atributos:

atributos globais, `autocomplete`, `autofocus`, `disabled`, `maxlength`, `minlength`, `name`, `placeholder`, `readonly` e `required`: funcionamento idêntico ao explicado na tag `<input>`.

**cols:** define o número de colunas do `<textarea>`, seu tamanho na horizontal (prefira *width* no CSS).

**rows:** determina o número de linhas do `<textarea>`, seu tamanho na vertical (prefira *height* no CSS).

Repare que `<textarea>` não possui o atributo *value*. O valor-padrão do campo deve ser contido dentro da tag (ou seja, entre `<textarea>` e `</textarea>`).

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <label for="obs">Observações</label><br>
      <textarea id="obs" rows="5" cols="50">Texto de
exemplo.</textarea>
    </form>
```

```
</body>
</html>
```

Código-fonte 47 – Exemplo de formulário usando <textarea>  
Fonte: Elaborado pelo autor (2016)

The image shows a web browser window with a form titled "Observações". Inside the form, there is a text area with the placeholder text "Texto de exemplo.". The text area is a simple rectangular box with a thin border. The form itself is a larger container with a light gray background.

Figura 25 – Exemplo de formulário usando <textarea>  
Fonte: Elaborado pelo autor (2016)

### 3.6 Tags <fieldset> e <legend>

Estas possibilitam um agrupamento de campos de um formulário.

Veja o exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <fieldset>
        <legend>Dados pessoais</legend>
        <label for="nome">Nome Completo:</label>
        <input type="text" id="nome"><br>
        <label for="idade">Idade:</label>
        <input type="number" id="idade"><br>
      </fieldset>
    </form>
  </body>
</html>
```

Código-fonte 48 – Exemplo de formulário usando <fieldset> e <legend>  
Fonte: Elaborado pelo autor (2016)





Dados pessoais

Nome Completo:

Idade:

Figura 26 – Exemplo de formulário usando `<fieldset>` e `<legend>`  
Fonte: Elaborado pelo autor (2016)

### 3.7 Tag `<datalist>`

Novo elemento de formulário, combina `<input type="text">` com `<select>`, ou seja, é uma caixa de seleção que permite digitação. Assim como `<select>`, utiliza `<option>` para determinar as opções disponíveis. Repare que o `<input type="text">` é necessário previamente utilizando o atributo `list`.

**ATENÇÃO:** A tag `<datalist>` não possui suporte em navegadores Safari, versão 12 e anteriores, e no Internet Explorer, na versão 9 e anteriores.

Eis o exemplo:

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <label for="navegador">Qual seu navegador
preferido?</label>
      <input list="navegadores" name="navegador">
      <datalist id="navegadores">
        <option value="Google Chrome">Google
Chrome</option>
        <option value="Mozilla Firefox">Mozilla
Firefox</option>
        <option value="Internet Explorer">Internet
Explorer</option>
        <option value="Opera">Opera</option>
```

```

        <option value="Apple Safari">Apple
        Safari</option>
      </datalist>
    </form>
  </body>
</html>

```

Código-fonte 49 – Exemplo de formulário usando <datalist>  
Fonte: Elaborado pelo autor (2016)

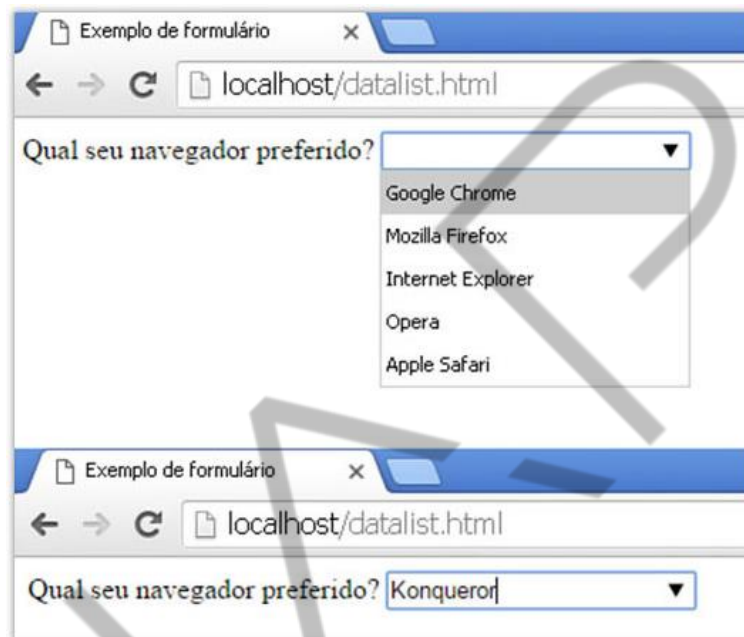


Figura 27 – Exemplo de formulário usando <datalist> em dois momentos: como caixa de seleção e digitando uma opção não encontrada na lista  
Fonte: Elaborado pelo autor (2016)

### 3.8 Tag <progress>

Novo recurso que possibilita mostrar o progresso de alguma ação. Geralmente utilizado para mostrar o progresso de um *upload*, usa os atributos *value* (valor atual) e *max* (valor total). Veja o exemplo:

```

<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <title>Exemplo de formulário</title>
    <meta charset="utf-8">
  </head>
  <body>
    <form id="formulario" method="POST">
      <progress value="25" max="100"></progress>
    </form>
  </body>
</html>

```

```
</form>
</body>
</html>
```

Código-fonte 50 – Exemplo de formulário usando <progress>  
Fonte: Elaborado pelo autor (2016)

**ATENÇÃO:** A tag <progress> não possui suporte em navegadores Internet Explorer versão 9 e anteriores.

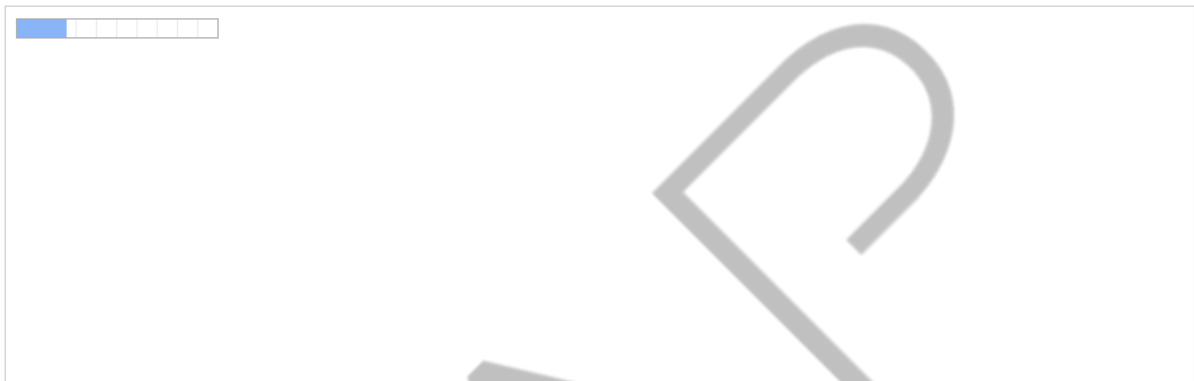


Figura 28 – Exemplo de formulário usando <progress>  
Fonte: Elaborado pelo autor (2016)

## CONCLUSÃO

O objetivo deste conteúdo foi abordar o básico e essencial da linguagem HTML; sendo assim, existem diversas *tags* e atributos que foram desconsiderados, seja por estarem descontinuados ou por terem pouco uso prático. Existem ainda outros usos para a *tag* `<input>` e *tags* importantes, como `<audio>` e `<video>`, que são abordados em nosso conteúdo específico de HTML 5, cuja leitura recomendamos.

EMANIP

## REFERÊNCIAS

W3C. **HTML 4.01 Specification**. Disponível em: <<https://www.w3.org/TR/html4/>>. Acesso em: 13 abr. 2021.

W3C. **HTML 5 Specification**. Disponível em: <<https://www.w3.org/TR/html5/>>. Acesso em: 13 abr. 2021.

W3C. **XHTML™ 1.0 The Extensible HyperText Markup Language**. 2. ed. Disponível em: <<https://www.w3.org/TR/xhtml1/>>. Acesso em: 13 abr. 2021.

W3C. **Extensible Markup Language (XML) 1.1**. 2. ed. Disponível em: <<https://www.w3.org/TR/xml11/>>. Acesso em: 13 abr. 2021.

W3Counter. **Browser & PlatformMarket Share**. Disponível em: <<http://www.w3counter.com/globalstats.php>>. Acesso em: 13 abr. 2021.