

PROGRAMAÇÃO FRONT END I

Diego César Batista Mariano



SOLUÇÕES
EDUCACIONAIS
INTEGRADAS



Trabalhando com *links* e formulários

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Desenvolver páginas HTML com *links* e âncoras.
- Desenvolver páginas HTML com formulários.
- Identificar os diferentes componentes de um formulário.

Introdução

Formulários permitem que usuários interajam com sistemas Web, estabelecendo conexão entre cliente e servidor.

Neste capítulo, você estudará sobre a interligação de páginas da Internet por meio de *links*. Será mostrado como desenvolver páginas HTML com *links* e âncoras, assim como com formulários. Você também será capaz de identificar os diferentes tipos de componentes de um formulário.

Desenvolvimento de páginas HTML com *links* e âncoras

HTML permite que páginas sejam interligadas por meio dos chamados *links* (SILVA, 2015). A *tag* âncora `<a>` é a responsável por isso, recebendo, no atributo `href`, o endereço para o qual o *link* deve apontar, conforme exemplo a seguir:

```
<a href="#">Eu sou um link</a>
```

Por padrão, a maior parte dos navegadores exibe *links* de texto sublinhados e na cor azul, exceto quando uma configuração diferente nas folhas de estilo CSS especifique outra configuração para a apresentação visual de um *link*.



Fique atento

Quando o atributo `href` recebe o valor `#`, ao clicar no *link*, nada acontece. Entretanto, ao deixar o valor do atributo `href` em branco, a página é recarregada.

Links relativos

Links relativos apontam para arquivos locais, ou seja, um arquivo presente no mesmo dispositivo do arquivo HTML em questão. Nos exemplos a seguir, você verá a criação de duas páginas HTML: página 1 e página 2. Ambas apresentam apenas um título indicando seu nome e um *link* para a página oposta.



Exemplo

```
<h1>Eu sou a página 1</h1>  
<a href="pagina2.html">Link para a página 2</a>
```



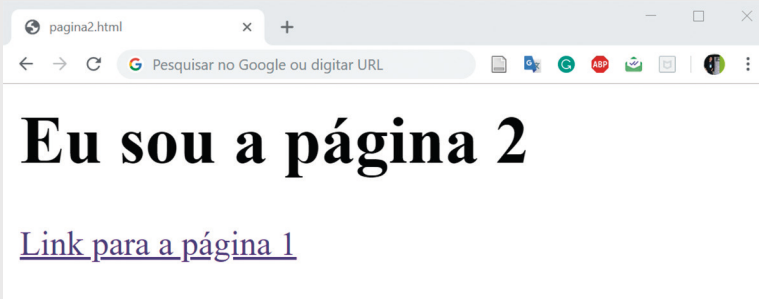
O *link* da página 1 recebe como atributo o valor `"pagina2.html"`. Observe que o valor deve ser exatamente igual ao nome do arquivo, incluindo a extensão `.html`. Nesse caso, o arquivo da página 2 deve estar no mesmo diretório do arquivo da página 1.

Observe, agora, a criação do arquivo “pagina2.html”.



Exemplo

```
<h1>Eu sou a página 2</h1>  
<a href="pagina1.html">Link para a página 1</a>
```



A página 2 apresenta um *link* que aponta para a página 1. Ao executar as páginas no navegador, é possível alternar de uma para outra, clicando nos *links* criados.



Fique atento

Links relativos podem receber como valor do atributo `href` endereços completos, como:

- C:\user\diego\desktop\pagina1.html (Windows)
- /user/diego/desktop/pagina1.html (MacOS X/Linux)

Além disso, é possível indicar *links* para acessar arquivos dentro de outros diretórios. No exemplo a seguir, foi criada uma pasta denominada “html”, para onde foi movido o arquivo “pagina2.html”. Veja como deveria ser o *link* no arquivo “pagina1.html”:

```
<a href="html/pagina2.html">Página 2</a>
```

Entretanto, no arquivo “pagina2.html”, o *link* para retornar à página 1 deve apontar para um diretório anterior. Para fazer isso, basta indicar que o *link* se encontra em um diretório anterior, adicionando “..” antes da barra:

```
<a href="../pagina1.html">Página 1</a>
```

Links absolutos

Links absolutos apontam para endereços externos. O valor de `href` deve conter um protocolo de acesso, como `http` ou `https` seguido por duas barras, e o endereço para o qual o *link* deve apontar.

```
<a href="https://www.google.com.br">Google</a>
```

Links para e-mails

Links também podem apontar para endereços de *e-mail*. Para isso, utilize a palavra `mailto` seguida de dois-pontos e um endereço de *e-mail*. Nesse caso, o navegador chamará o programa-padrão de envio de *e-mails* do computador, informando que a mensagem deve ser enviada ao *e-mail* indicado.

```
<a href="mailto:peessoa@sagah.com.br">Contato por e-mail</a>
```

Links em imagens

Links podem ser combinados com imagens. Nesse caso, o código HTML da imagem deve ser inserido dentro da tag âncora `<a>` no lugar do texto.

```
<a href="#">  
    
</a>
```

Atributo `target`

O atributo `target` indica o contexto de navegação que determinado *link* deve carregar (nova aba, nova janela ou *frame*). No Quadro 1, a seguir, estão os valores que podem ser atribuídos ao *target*.

Quadro 1. Valores que podem ser atribuídos ao *target*

Valor	Definição
<code>_self</code>	Carrega o <i>link</i> na aba atual (valor padrão utilizado quando o atributo não é definido).
<code>_blank</code>	Carrega o <i>link</i> em uma nova aba.
<code>_parent</code>	Carrega o <i>link</i> no <i>frame</i> pai.
<code>_top</code>	Carrega o <i>link</i> no corpo inteiro da janela.

Desenvolvimento de páginas HTML com formulários

Nos primeiros anos da Internet, as páginas Web eram estáticas, permitindo ao usuário apenas a leitura de informações. Com a evolução das tecnologias e linguagens de desenvolvimento Web, foi possível que usuários não apenas lessem, mas, também, enviassem informações. Um exemplo, disso são os chamados formulários (Figura 1).

Formulários permitem que usuários interajam com sistemas Web (MARIANO; MELO-MINARDI, 2017), estabelecendo conexão entre cliente e servidor, por meio do envio de informações a serem processadas por linguagens de programação *back-end*.

```
<div class="form-box login-box active">
  <form class="form-signin">
    <input type="text" name="username"
      Email address" value="{{username}}"/>
    <input type="password" name="password"
      placeholder="password" value="{{password}}"/>
    <input type="checkbox" id="login-remember"/>
    <label for="login-remember">Keep me logged in</label>
    <button class="btn btn-tall btn-wide" type="submit">Log In</button>
  </form>
  {{#error}}
  <div class="error-message">{{error}}</div>
</div>
```

Figura 1. Criando formulários HTML.

Fonte: photovibes/Shutterstock.com.

Formulários têm muitas utilidades, como em páginas de contato, que permitem que o usuário do *site* envie informações para o criador (Figura 2).

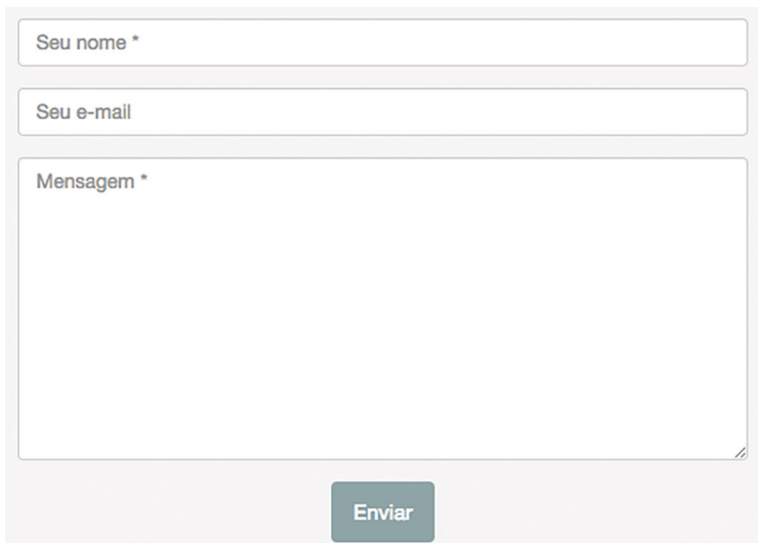
A imagem mostra um exemplo de formulário web para envio de e-mail. O formulário é composto por três campos de entrada: um campo de texto para "Seu nome *" com uma borda arredondada e uma seta de foco; um campo de texto para "Seu e-mail" com uma borda arredondada e uma seta de foco; e um campo de texto de área para "Mensagem *" com uma borda arredondada e uma seta de foco. Abaixo dos campos, há um botão de envio com o texto "Enviar" em um fundo cinza escuro e letras brancas.

Figura 2. Exemplo de formulário para envio de *e-mail*.

Fonte: Mariano e Melo-Minardi (2017).

Tag <form>

Formulários podem ser criados com a *tag* <form>, que recebe diversos atributos (<FORM>..., 2019), dentre eles:

- **name** — nome que identifica o formulário;
- **action** — endereço para onde o formulário deve ser enviado;
- **method** — post ou get (Quadro 2);
- **enctype** — valor usado como identificador de duas partes para formatos transmitidos na Internet (usado para o método post). Pode receber os valores: `application/x-www-form-urlencoded` (valor padrão), `multipart/form-data` (para envio de arquivos) e `multipart/form-data` (para HTML5).

No exemplo a seguir, você verá como um formulário é criado. O atributo `action` recebe a página para a qual o formulário será enviado (aqui denominada como “`processa_formulario.php`”). O método utilizado foi o `post`, e o nome do formulário é “`meu_formulario`”.

```
<form action="processa_formulario.php" method="post"
name="meu_formulario">
    [...]
</form>
```

Quadro 2. Métodos POST e GET

GET	No método GET, as informações do formulário são transmitidas diretamente pela URL. Em geral, o método GET permite envio de <i>strings</i> até 255 caracteres.
POST	O método POST encapsula as informações e as envia junto ao corpo da requisição HTTP.

Diferentes componentes de um formulário

Formulários são componentes HTML que recebem informações. A principal *tag* utilizada por um formulário é a *tag* `<input>`, que contém diversos tipos. Além disso, formulários também podem receber campos como `<button>` ou, até mesmo, `<label>` para identificar rótulos.

Tag `input`

A *tag* `<input>` permite a inserção de dados em formulários e pode receber como atributos:

- `name` — nome do formulário que será usado para processamento em linguagens *back-end*;
- `value` — atributo que recebe o valor-padrão a ser exibido no formulário;
- `placeholder` — texto que aparece dentro de caixas de texto;
- `type` — identificação do tipo de campo;

O Quadro 3, a seguir, mostra os tipos de *input* de formulários.

Quadro 3. Tipos de *input* de formulários

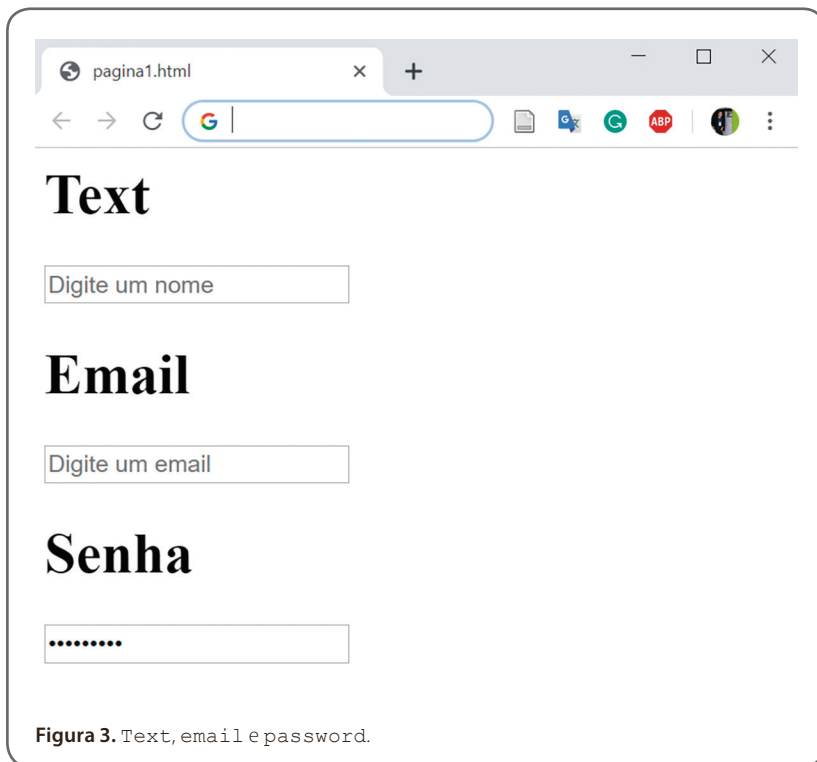
Valor	Definição
<code>text</code>	Caixa de texto simples
<code>email</code>	Caixa de texto com verificação de <i>e-mail</i>
<code>password</code>	Campo de senha
<code>checkbox</code>	Caixa de seleção múltipla
<code>radio</code>	Caixa de seleção única
<code>file</code>	Arquivo
<code>submit</code>	Botão de submissão de formulário
<code>hidden</code>	Campo oculto
<code>button</code>	Botão simples

Tipo text, email e password

Os campos text, email e password são bastante parecidos (Figura 3). A principal diferença é que email verifica se o valor digitado corresponde a um *e-mail* (busca pelo símbolo de @), e o password criptografa os valores digitados (oculta caracteres).

```
<form action="#" method="post" name="formulario">
  <h1>Text</h1>
  <p>
    <input type="text" name="texto" placeholder="Digite um
nome">
  </p>
  <h1>Email</h1>
  <p>
    <input type="email" name="email" placeholder="Digite um
e-mail">
  </p>
  <h1>Senha</h1>
  <p>
    <input type="password" name="senha" placeholder="Digite
a senha">
  </p>
</form>
```

O atributo placeholder exibe uma mensagem que desaparece quando se clica no campo.



Tipo checkbox e radio

checkbox e radio são dois tipos de campos de marcação (Figura 4). A principal diferença entre eles é que grupos de campos checkbox podem ser marcados múltiplas vezes, enquanto os de radio só permitem a escolha de um.

```
<form action="#" method="post" name="formulario">
  <h1>Checkbox</h1>
  <p>
    <input type="checkbox" name="nome_unico1" value="1" />
    <label>1</label>
  </p>
  <p>
    <input type="checkbox" name="nome_unico1" value="2" />
    <label>2</label>
  </p>
```

```
<h1>Radio</h1>
<p>
  <input type="radio" name="nome_unico2" value="1" />
  <label>1</label>
</p>
<p>
  <input type="radio" name="nome_unico2" value="2" />
  <label>2</label>
</p>
</form>
```

Para que sejam identificados como um grupo, devem receber o mesmo valor no campo name.

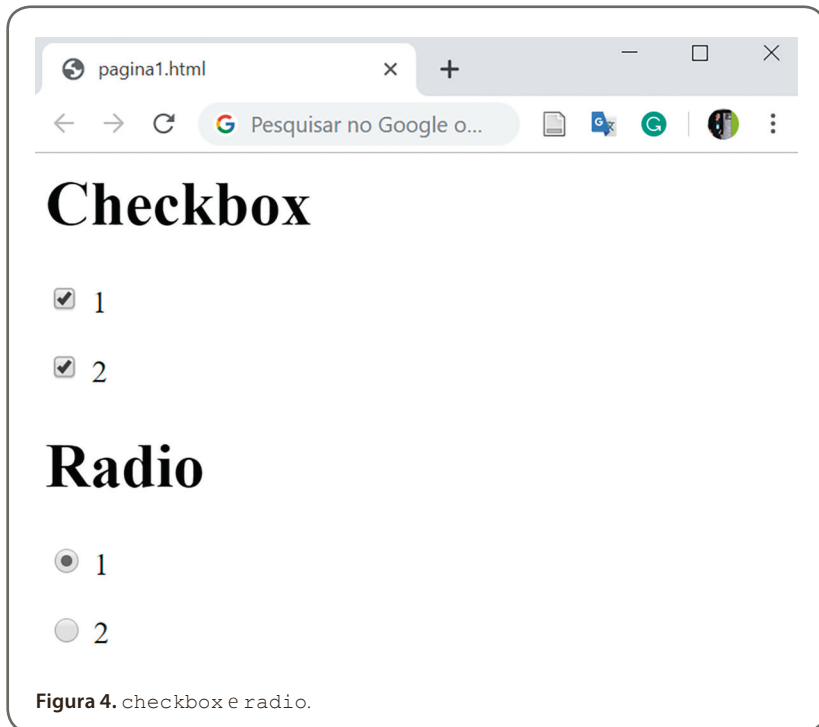


Figura 4. checkbox e radio.

Tipos `hidden` e `file`

Campos do tipo `hidden` ocultam informações (Figura 5). Podem ser utilizados para armazenar códigos de identificação ou qualquer outra informação que não deve ser exibida para o usuário.

O tipo `file` permite o envio de arquivos.

```
<form action="#" method="post" name="formulario">
  <h1>Arquivo</h1>
  <p>
    <input type="file" name="arquivo">
    <input type="hidden" value="codigo _ secreto _ 123">
  </p>
</form>
```

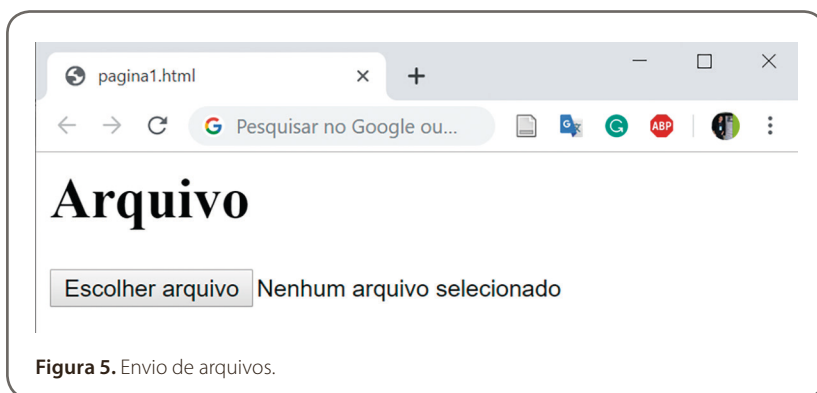


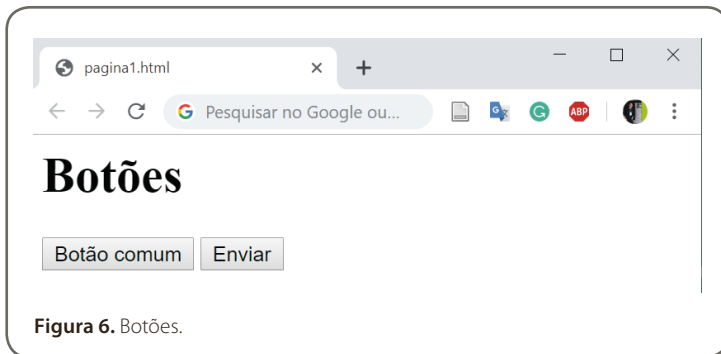
Figura 5. Envio de arquivos.

Tipos `button` e `submit`

Botões `button` e `submit` permitem a execução de ações (Figura 6). Entretanto, o botão do tipo `submit` é especial, pois é por meio dele que o formulário é enviado.

```
<form action="#" method="post" name="formulario">
  <h1>Botões</h1>
  <p>
    <input type="button" value="Botão comum">
    <input type="submit" value="Enviar">
  </p>
</form>
```

O atributo `value` é quem define o valor do texto que será exibido no botão. Ao clicar em enviar, a página é redirecionada para onde o atributo `action` do `<form>` indicar.

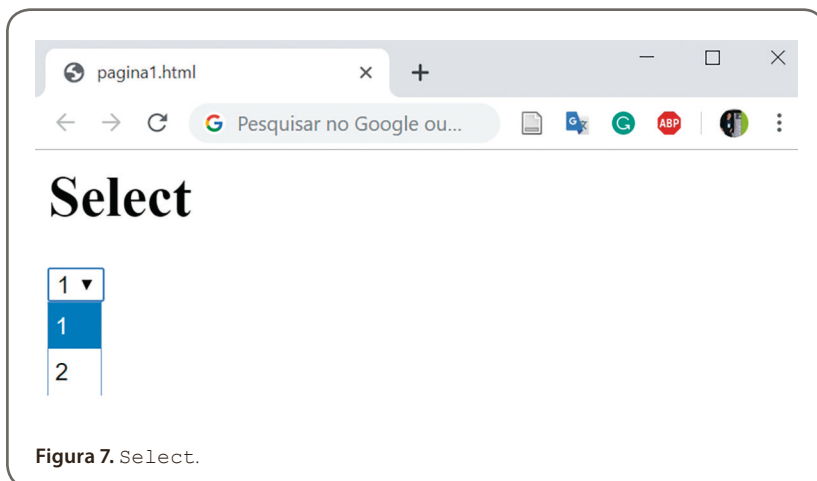


Tag `<select>`

As tags `<select>` (Figura 7) permitem que o usuário selecione entre uma série de opções determinadas pelas tags `<option>`.

```
<form action="#" method="post" name="formulario">
  <h1>Select</h1>
  <p>
    <select name="selecione">
      <option value="1">1</option>
      <option value="2">2</option>
    </select>
  </p>
</form>
```

Observe que a *tag* `<option>` recebe o atributo `value` que define o valor a ser enviado, seguido por um valor que será mostrado para o usuário.



Tag `<textarea>`

Tags do tipo `<textarea>` (Figura 8) são parecidas com `<input>` do tipo `text`, servindo para receber valores em texto. A principal diferença é que `<textarea>` permite que o valor da caixa seja alterado.

```
<form action="#" method="post" name="formulario">
  <h1>Textarea</h1>
  <p>
    <textarea cols="50" rows="10"></textarea>
  </p>
</form>
```

O tamanho de áreas de texto pode ser alterado por meio dos atributos `cols` (número de colunas) e `rows` (número de linhas).

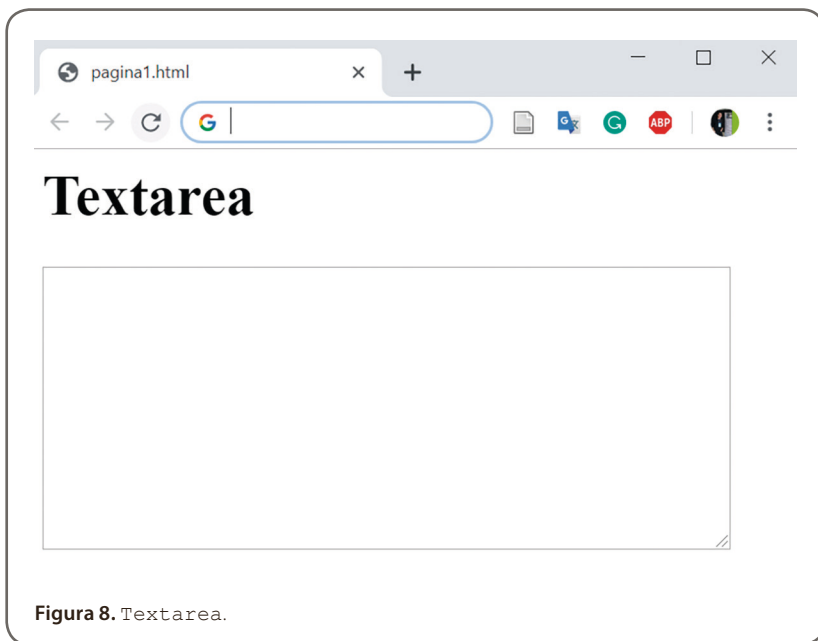


Figura 8. Textarea.



Referências

<FORM> – HTML: Linguagem de Marcação de Hipertexto. *MDN Web Docs*, Mountain View, 2019. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTML/Element/form>. Acesso em: 28 set. 2019.

MARIANO, D.; MELO-MINARDI, R. *Introdução à programação web para bioinformática: HTML, CSS, PHP & JavaScript*. North Charleston: CreateSpace Independent Publishing Platform, 2017. 410 p.

SILVA, M. S. *Fundamentos de HTML5 e CSS3*. São Paulo: Novatec, 2015. 304 p.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:



SOLUÇÕES
EDUCACIONAIS
INTEGRADAS