

ntander Coders 2024.1 | Front-End | #1178

Isaias Soares

Id: 1178008



Formulários

Tudo que fizemos até agora foi para mostrar conteúdo na tela, nada era capaz de receber conteúdo inserido pela pessoa que está usando a página.

Formulários são o mecanismo do HTML para entrada de dados.

Para fazer um formulário, usamos como raiz de todos os elementos a *tag* `<form></form>`.

Dentro dela, podemos colocar os elementos HTML que vimos antes para fazer as divisões, mas também podemos colocar uma série de elementos HTML de entrada de dados.

Esses elementos são campos de texto, caixas de seleção, *checkboxes*, *radio buttons*, entre outros...

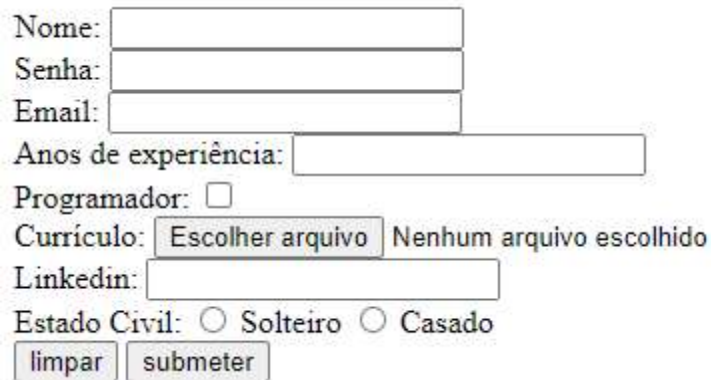
Inputs

Grande parte deles são criados pelo elemento HTML `<input/>`, e no seu atributo `type`, você pode decidir qual tipo de *input* deseja. Vejamos:

Exemplo: Campos de input

```
<form>
  <div>Nome: <input name="nome" type="text" /></div>
  <div>Senha: <input name="senha" type="password" /></div>
  <div>Email: <input name="email" type="email" /></div>
  <div><input name="escondido" type="hidden" value="form-rh-2020" /></div>
  <div>Anos de experiência: <input name="xp" type="number" /></div>
  <div>Programador: <input name="programador" type="checkbox" /></div>
  <div>Currículo: <input name="curriculum" type="file" /></div>
  <div>Linkedin: <input name="linkedin" type="url" /></div>
  <div>
    Estado Civil:
    <input name="estadocivil" type="radio" value="solteiro" /> Solteiro
    <input name="estadocivil" type="radio" value="casado" /> Casado
  </div>
  <div>
    <input name="reset" type="reset" value="limpar" />
    <input name="submit" type="submit" value="submeter" />
  </div>
</form>
```

Resultado:



Nome:

Senha:

Email:

Anos de experiência:

Programador: ☐

Currículo: Nenhum arquivo escolhido

LinkedIn:

Estado Civil: ☐ Solteiro ☐ Casado

Imagem 1 - Formulário (Fonte da imagem: do autor)

Observe que o mesmo elemento `input` tornou-se campos de texto, *checkboxes*, *radios* (bolinha de múltipla escolha), botões de escolher arquivos, até mesmo os botões de limpar e enviar o formulário. Ele é a nossa principal ferramenta de inserção de dados. Fizemos até um `input` que não aparece na página usando `type="hidden"`, onde você pode colocar seus próprios dados...

Através da imagem acima, você não conseguirá ver isso, mas cada campo se comporta de uma maneira diferente. O campo de experiência, por exemplo, tem botões para cima e para baixo para incrementar e decrementar o valor. O campo de senha não mostra o que está escrito nele, o do LinkedIn é um campo de url e por aí vai.

Vale a pena colocar esse código em uma página HTML e abrir no navegador para ter a experiência completa...

O atributo `name` é extremamente importante para os elementos dentro de um formulário, pois será a partir dele que o servidor poderá compor as informações em pares chave-valor, para que possamos acessá-la depois. Sendo a chave o `name` e valor o que o usuário digitou no campo ou a propriedade `value`.

Todos os `inputs` devem ter um `name` único para poderem ser processados mais tarde (muito cuidado ao copiar e colar!), a única exceção é o *radio button*, que, por ser um elemento de múltipla escolha, deve usar em todas as opções o mesmo `name`, o que você recebe depois é o `value` do *radio* escolhido pelo usuário (então não esqueça dos *values*!).

Selects (caixas de seleção)

Além dos *inputs*, temos mais dois elementos muito utilizados nos formulários, o primeiro deles é a caixa de seleção, que te permite pré-definir opções para que o usuário escolha.

Exemplo: Caixa de Seleção

```
<select name="curso">
  <option>Selecione...</option>
  <option value="WFR">Web Frontend & React</option>
  <option value="WFS">Web Fullstack</option>
  <option value="WRD">Web & React (Digital)</option>
</select>
```

Resultado:

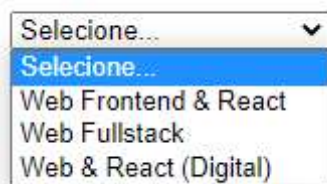


Imagem 2 - Caixa de seleção - Select (Fonte da imagem: do autor)_

Observe que o `name` vai no `select` enquanto os `values` vão nos `option`. Apenas o `value` do `option` selecionado vai ser enviado ao servidor.

Se quiser pré-selecionar um `option`, basta adicionar o atributo `selected` nela. *Checkboxes* e *radio buttons* têm um equivalente chamado `checked`.

Via de regra, sempre faço um primeiro `option` dizendo "Selecione", para não arriscar que o usuário passe pelo campo sem alterá-lo e envie sem querer a primeira opção. Veja que esse "Selecione" não tem o atributo `value`, assim posso fazer uma validação prévia se o campo foi ou não alterado.

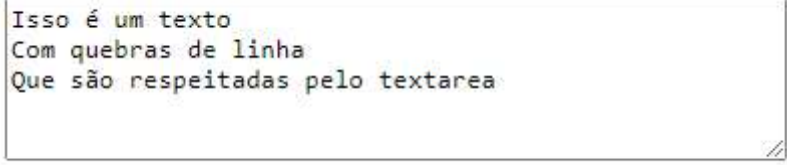
TextArea (campos de texto de múltiplas linhas)

O segundo elemento que não é um `input` é o `<textarea></textarea>`. Ele serve como caixa de texto também, mas permite que você quebre linhas para adicionar textos grandes.

Exemplo: TextArea

```
<textarea name="texto" cols="50" rows="5">  
    Isso é um texto  
    Com quebras de linha  
    Que são respeitadas pelo textarea  
</textarea>
```

Resultado:



```
Isso é um texto  
Com quebras de linha  
Que são respeitadas pelo textarea
```

Imagem 3 - Textarea (Fonte da imagem: do autor)

Observe que as quebras de linha são preservadas no texto original, e que usamos os atributos `cols` e `rows` para determinar o tamanho da caixa de texto baseado em linhas e caracteres. No caso, dissemos que ela terá 5 linhas de 50 caracteres cada.

As caixas de texto podem ser redimensionadas usando as duas marquinhas no canto inferior direito. `cols` e `rows` são apenas os valores iniciais.

Ela quebrará linhas automaticamente quando o texto chegar ao limite, mais isso não adiciona um caractere de quebra de linha. Pressionar `enter` adiciona o caractere de quebra de linha. Os espaços também são preservados.

Nos formulários, pressionar `enter` em um campo submete o formulário, no caso das caixas de texto esse comportamento não acontece.

Submissão do formulário

Afinal, quando submeto um formulário, para onde ele vai?

É você quem decide! A tag `form` tem um atributo especial chamado `action`; nele, você coloca a URL de um recurso do servidor capaz de processar os formulários. Por exemplo:

```
<form action="./processa-form-rh.php" method="POST">...</form>
```

Nesse exemplo, estamos dizendo que o formulário deve ser enviado para um script PHP chamado `processa-form-rh.php` que obterá os dados preenchidos e fará o que é necessário para processá-lo. Também passamos o `method` como sendo `POST` isso quer dizer que os dados vão no corpo da requisição e não na URL para o servidor.

Sempre use `POST`.

Colocamos PHP como exemplo, mas cada linguagem web te oferecerá uma forma de produzir um recurso no servidor capaz de processar o formulário. Algumas linguagens/frameworks usam rotas em vez de arquivos, como ASP.NET MVC e Express (node.js). Nelas, podemos ter urls mais amigáveis como `action="/rh/novoCandidato"`.

Referências e Links úteis

- [Meu primeiro formulário HTML](#). MDN Web Docs, 02/08/2021. Acesso em: 27/06/2022.

Próximo Tópico