Formulários

Os formulários são uma parte importante de uma página web, eles são utilizados para coletar informações dos usuários.

Permite que dados sejam inseridos, como texto, números, datas, arquivos, seleções, etc.

Temos diversos tipos de campos de formulários, como `<input>`, `<textarea>`, `<select>`, `<button>`, `<label>`, dentre outros.

pedro.mateus@unicesumar.edu.br

Os inputs são os elementos mais comuns de um formulário, e possuem diversos tipos, como `text`, `password`, `email`, `number`, `date`, `file`, `checkbox`, `radio`, etc. Eles são definidos através do atributo `type`.

```
<!-- Para entrada de texto longo (descrição, comentário). -->
<input type="textarea" />
```

Além do type, também existem outros atributos importantes para os inputs, como `name`, `value`, `placeholder`, `required`, `disabled`, `readonly`, `min`, `max`, etc.

O atributo name é utilizado para identificar o campo, e é o nome que será enviado junto aos dados formulário for submetido.

```
<input type="textarea" name="descricao" />
```

O atributo value é um atributo opcional que define o valor inicial do campo. É obrigatório para inputs do tipo `checkbox` e `radio`. Pode ser utilizado via javascript para definir ou obter o valor do campo.

```
<input type="textarea" name="descricao" value="Sua descricao" />
```

```
document.querySelector('input[name="descricao"]').value = 'Nova descrição'
```

Todo formulário possui uma estrutura básica, que é composta por um elemento `<form>`, que é o container do formulário, e os elementos de input, botões, etc.

```
<!-- Formulário que envia os dados para /submit
     ao clicar no botão "Enviar" -->
<form name="meu-form" action="/rota/submit">
  <input type="text" name="nome" />
  <input type="email" name="email" />
  <button type="submit">Enviar</button>
</form>
```

O button com type `submit` indica que o formulário será submetido ao clicar no botão. O atributo `action` define a rota para onde os dados do formulário serão enviados.

Também é possível utilizar o atributo `method` para definir o método de envio dos dados, que pode ser `get` ou `post`.

```
<!-- Formulário que envia os dados para /submit
     ao clicar no botão "Enviar" -->
<form name="meu-form" action="/rota/submit" method="post">
  <input type="text" name="nome" />
  <input type="email" name="email" />
  <button type="submit">Enviar</button>
</form>
```

Agora será chamada a rota `/submit` com o método `post` ao submeter o formulário.

Para páginas estáticas, podemos optar por não utilizar o atributo `action`, e utilizar o atributo `onsubmit` para definir uma função que será chamada ao submeter o formulário.

```
<!-- Formulário que chama a função "enviar"
     ao clicar no botão "Enviar" -->
<form name="meu-form" onsubmit="enviar()">
  <input type="text" name="nome" />
  <input type="email" name="email" />
  <button type="submit">Enviar</button>
</form>
```

```
function enviar() {
  alert('Formulário enviado')
}
```

Conforme demonstrado nos exemplos acima, o elemento `button` tem um papel importante nos formulários e o atributo type define o comportamento do botão.

O type `submit` é utilizado para submeter o formulário, o type `reset` é utilizado para limpar os campos do formulário, e o type `button` é utilizado para botões que não submetem o formulário. **O type submit também é disparado ao pressionar a tecla `enter` em um campo de texto.**

Outro elemento importante para os formulários é o label. Nos exemplos acima podemos notar que os inputs não possuem um texto associado, o que pode ser ruim para a acessibilidade e usabilidade.

Podemos definir uma label para um input utilizando o atributo `for` do label, que recebe o id do input. Formando uma associação entre o label e o input. Dessa maneira temos uma página mais acessível e com uma melhor usabilidade. Vide que ao clicar no texto do label, o input é selecionado, e leitores de tela conseguem identificar a associação.

```
<!-- Formulário com labels associadas aos inputs -->
<form name="meu-form" onsubmit="enviar()">
  <label for="nome">Nome</label>
  <input type="text" name="nome" id="nome" />

  <label for="email">Email</label>
  <input type="email" name="email" id="email" />

  <button type="submit">Enviar</button>
</form>
```

Podemos também utilizar o atributo `placeholder` para adicionar um texto de exemplo ao input, que será exibido quando o input estiver vazio. Indicando ao usuário o que deve ser inserido no campo, ou até mesmo o formato que deve ser seguido, com um exemplo.

```
<!-- Formulário com placeholders nos inputs -->
<form name="meu-form" onsubmit="enviar()">
  <label for="nome">Nome</label>
  <input type="text" name="nome" id="nome" placeholder="Digite seu nome" />

  <label for="email">Email</label>
  <input type="email" name="email" id="email" placeholder="email@email.com" />

  <button type="submit">Enviar</button>
</form>
```

Outro atributo comum é o `required`, que é utilizado para definir que um campo é obrigatório, e o formulário não pode ser submetido se o campo estiver vazio. Ao tentar submeter o formulário com um campo obrigatório vazio, o navegador exibirá uma mensagem de erro, informando que o campo precisa ser preenchido.

```html
<!-- Formulário com campos obrigatórios -->
<form name="meu-form" onsubmit="enviar()">
  <label for="nome">Nome</label>
  <input
    type="text"
    name="nome"
    id="nome"
    placeholder="Digite seu nome"
    required
  />

  <label for="email">Email</label>
  <input
    type="email"
    name="email"
    id="email"
    placeholder="email@email.com"
    required
  />

  <button type="submit">Enviar</button>
</form>
```

Outro atributo comum é o `disabled`, que é utilizado para desabilitar um campo, impedindo que o usuário interaja com ele. O campo desabilitado não será enviado junto com os dados do formulário.

```
<!-- Formulário com campos desabilitados -->
<input
  type="text"
  name="nome"
  id="nome"
  placeholder="Digite seu nome"
  disabled
/>
```

Customizando formulários

```css
/* Estilo padrão para os inputs */
input {
  padding: 10px;
  margin: 10px;
  border: 1px solid #ccc;
}

/* Estilo para os inputs válidos */
input:valid {
  border-color: green;
}

/* Estilo para os inputs inválidos */
input:invalid {
  border-color: red;
}

/* Estilo para os inputs em foco */
input:focus {
  border-color: blue;
}

/* Estilo para os inputs desabilitados */
input:disabled {
  background-color: lightgrey;
}
```

Adicionando lógica ao formulário

Por padrão os formulários recarregam a página ao serem submetidos, para evitar esse comportamento, podemos utilizar o método `preventDefault()` do evento, que previne o comportamento padrão do evento. Primeiro temos que passar o evento como parâmetro da função.

```
<form name="meu-form" onsubmit="enviar(event)">[...]</form>
```

```
function enviar(e) {
  e.preventDefault()
}
```

Podemos acessar os dados do formulário através do objeto `FormData`, que é um objeto que permite a criação de pares chave/valor a partir de um formulário.

```
function enviar(e) {
  e.preventDefault()

  const form = e.target
  const formData = new FormData(form)

  console.log(formData.get('nome'))
  console.log(formData.get('email'))
  console.log(formData.get('email'))
}
```