ESCOLA ESTADUAL PROFESSORA SEBASTIANA DE ALMEIDA E SILVA
Rua São José, nº 30, Jaguaruçu - MG. CEP: 35.188-000 Tel: (31) 3845-1209

DISCIPLINA: Front End I	**PROFESSOR(A):** Prentys Assis
ATIVIDADE: Formulários em HTML: Estrutura, Tipos e Boas Práticas	**DATA:** ____/____/____
ALUNO(A):	**TURMA:** 2º Ano

FORMULÁRIOS EM HTML

Os formulários são a principal forma de interação entre usuário e aplicação web. Eles permitem a coleta de dados que podem ser enviados a um servidor para processamento, como em cadastros, logins, pesquisas e muito mais.

1. A Estrutura Básica de um Formulário

Um formulário HTML começa com a tag `<form>` e normalmente possui ao menos dois atributos importantes:

- `action` : endereço (URL) para onde os dados serão enviados.
- `method` : método HTTP usado para o envio (GET ou POST).

```
<form action="/envia" method="post">
  <label for="nome">Nome: </label>
  <input type="text" id="nome" name="nome" required>

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

Os formulários são a ponte entre o navegador e o servidor: eles organizam dados em pares `name/value` que serão processados pela aplicação no backend. Para buscas ou filtros, `GET` é conveniente por deixar os parâmetros visíveis na URL. Para cadastros ou atualizações que alteram dados no servidor, `POST` é o método mais apropriado e seguro.

2. Labels e Acessibilidade

A tag `<label>` conecta texto descritivo a um campo de formulário através do atributo `for` , que deve corresponder ao `id` do campo. Isso melhora a acessibilidade, permitindo que leitores de tela identifiquem os campos e que usuários cliquem no texto para focar no input correspondente.

```
<label for="email">E-mail</label>
<input type="email" id="email" name="email" placeholder="seu@exemplo.com">
```

O uso de `placeholder` é útil como dica, mas ele desaparece quando o usuário começa a digitar. Por isso, ele nunca deve substituir uma `<label>` como única forma de identificação do campo.

3. Caixas de Texto e Senha

Os campos de entrada de texto mais comuns são:

- `type="text"` : Para texto livre.
- `type="password"` : Oculta o texto digitado, ideal para senhas.

Atributos como `maxlength` e `minlength` controlam o tamanho do texto inserido.

```
<label for="usuario">Usuário</label>
<input type="text" id="usuario" name="usuario" maxlength="30" required>

<label for="senha">Senha</label>
<input type="password" id="senha" name="senha" minlength="6" required>
```

Para senhas, considere oferecer um botão para revelar/ocultar o conteúdo. Validações no navegador ajudam na experiência, mas a validação no servidor é indispensável para a segurança.

4. Tipos Numéricos e Datas

O HTML5 introduziu vários tipos de `input` que melhoram a usabilidade, especialmente em dispositivos móveis, ao exibirem teclados e controles nativos apropriados.

- `number` : Para números, com atributos `min` , `max` e `step` .
- `range` : Controle deslizante (slider).
- `date` , `time` , `month` , `week` , `datetime-local` : Seletores de data e hora.

```
<label for="idade">Idade</label>
<input type="number" id="idade" name="idade" min="0" max="120">

<label for="data">Data de nascimento</label>
<input type="date" id="data" name="nascimento">
```

A aparência desses controles pode variar entre navegadores, então use textos e placeholders para comunicar restrições importantes. O `input type="date"` retorna uma string no formato AAAA-MM-DD.

5. Telefone e E-mail

Utilizar tipos semânticos ajuda na validação automática do navegador e no preenchimento automático.

- `type="email"` : Valida o formato básico de um e-mail (presença do @ e domínio).
- `type="tel"` : Indica a intenção de um número de telefone, otimizando o teclado em dispositivos móveis.

```
<label for="email">E-mail</label>
<input type="email" id="email" name="email" required>

<label for="tel">Telefone</label>
<input type="tel" id="tel" name="telefone" placeholder="(99) 99999-9999">
```

O `type="tel"` não possui uma validação estrita por padrão. Para forçar formatos específicos, combine-o com o atributo `pattern` ou utilize máscaras via JavaScript.

6. Checkbox e Radio

Esses tipos são usados para seleções:

- `checkbox` : Permite múltiplas seleções independentes.
- `radio` : Permite apenas uma escolha dentro de um grupo com o mesmo atributo `name` .

```
<fieldset>
  <legend>Quais linguagens você conhece?</legend>
```

```
  <label><input type="checkbox" name="linguagens" value="html"> HTML</label>
  <label><input type="checkbox" name="linguagens" value="css"> CSS</label>
  <label><input type="checkbox" name="linguagens" value="js"> JavaScript</label>
</fieldset>

<fieldset>
  <legend>Você já programou antes?</legend>
  <label><input type="radio" name="experiencia" value="sim"> Sim</label>
  <label><input type="radio" name="experiencia" value="nao"> Não</label>
</fieldset>
```

Agrupar campos relacionados com `<fieldset>` e `<legend>` melhora a semântica e a acessibilidade do formulário.

7. Select, Datalist e Textarea

- `<select>` : Cria um menu suspenso (dropdown) com uma lista de `<option>` .
- `<datalist>` : Fornece sugestões de autopreenchimento para um `<input>` , permitindo que o usuário digite um valor diferente.
- `<textarea>` : Cria um campo de texto de múltiplas linhas.

```
<label for="pais">País</label>
<select id="pais" name="pais">
  <option value="br">Brasil</option>
  <option value="pt">Portugal</option>
</select>

<label for="cidade">Cidade (sugestões)</label>
<input list="cidades" id="cidade" name="cidade">
<datalist id="cidades">
  <option value="São Paulo">
  <option value="Rio de Janeiro">
</datalist>

<label for="msg">Mensagem</label>
<textarea id="msg" name="mensagem" rows="6" placeholder="Escreva sua mensagem..."></textarea>
```

8. Upload de Arquivos e Enctype

Para permitir que os usuários enviem arquivos, use `<input type="file">` e configure o atributo `enctype` do formulário como `"multipart/form-data"` .

```
<form action="/upload" method="post" enctype="multipart/form-data">
  <label for="arquivo">Escolha um arquivo</label>
  <input type="file" id="arquivo" name="arquivo">
  <button type="submit">Enviar</button>
</form>
```

Sempre valide o tipo e o tamanho dos arquivos no lado do servidor por segurança. O atributo `accept` no input (ex: `accept="image/*"`) pode ser usado para sugerir os tipos de arquivo permitidos.

9. Botões e Tipos de Button

A tag `<button>` é mais flexível que `<input type="button">` , pois pode conter HTML. Seus tipos principais são:

- `type="submit"` (padrão): Envia os dados do formulário.
- `type="reset"` : Limpa todos os campos do formulário para seus valores iniciais.
- `type="button"` : Botão genérico, sem ação padrão, geralmente controlado por JavaScript.

Evite usar `type="reset"` em formulários importantes, pois os usuários podem clicar acidentalmente e perder os dados digitados.

10. Validação e Atributos Importantes

O HTML oferece atributos para validação no lado do cliente, fornecendo feedback imediato ao usuário.

- `required` : Torna o campo obrigatório.
- `pattern` : Valida o valor do campo contra uma expressão regular (regex).
- `min` , `max` , `step` : Para campos numéricos e de data.
- `readonly` , `disabled` : Para controlar a interação com o campo.

```
<label for="cpf">CPF</label>
<input type="text" id="cpf" name="cpf" pattern="\d{3}\.\d{3}\.\d{3}-\d{2}" placeholder="000.000.000-00" required>
```

Lembre-se: a validação no cliente (navegador) melhora a experiência, mas a validação no servidor é essencial para a segurança e integridade dos dados.

11. Boas Práticas

- Sempre use o atributo `name` em todos os campos que precisam ser enviados ao servidor.
- Prefira sempre usar `<label>` para melhorar a acessibilidade.
- Agrupe controles relacionados em `<fieldset>` .
- Evite usar `placeholder` como a única forma de instrução.
- Teste seus formulários em dispositivos móveis para garantir uma boa experiência de usuário.