

JavaScript: Fundamentos para Desenvolvimento Web Interativo - Turma 2025B

6.2.1 Atributos de validação embutidos do HTML5

O HTML5 introduziu vários atributos de validação que podem ser utilizados para validar dados de forma nativa, sem necessidade de escrever qualquer código JavaScript. Esses atributos são úteis para verificar se os campos obrigatórios foram preenchidos, se os valores seguem padrões predefinidos ou se atendem a certos tipos de entrada. Alguns dos atributos mais usados incluem:

- **Obrigatoriedade (required):** Indica que o campo não pode ficar vazio.
- **Comprimento (minlength, maxlength):** Define o comprimento mínimo e/ou máximo dos dados.
- **Tipo (type):** Especifica o tipo de entrada, como **email**, **number**, ou **url**.
- **Padrão (pattern):** Estabelece um padrão de expressão regular que o valor deve corresponder.

Neste contexto, se os dados inseridos seguirem todas as regras especificadas, eles são considerados válidos; caso contrário, são marcados como inválidos, de maneira que:

- **Quando um elemento é válido:** O elemento corresponde à pseudo-classe CSS `:valid`, permitindo que você aplique estilos específicos a elementos válidos. Se o usuário tentar enviar os dados, o navegador enviará o formulário, desde que não haja outra restrição (como validação adicional com JavaScript).
- **Quando um elemento é inválido:** O elemento corresponde à pseudo-classe CSS `:invalid`, permitindo que você estilize elementos inválidos para chamar a atenção do usuário. Se o usuário tentar enviar o formulário, o navegador impedirá a submissão e exibirá automaticamente uma mensagem de erro para indicar ao usuário qual correção precisa ser feita.

Por exemplo, considere as seguintes validações HTML5:

```
<input type="text" id="userName" name="userName" minlength="4" maxlength="15" pattern="[A-Za-z]+" required>  
<input type="email" id="userEmail" name="userEmail" placeholder="Digite seu email" required>
```

A primeira validação está verificando a obrigatoriedade de preenchimento do campo de nome, definindo um tamanho mínimo e máximo de caracteres e estabelecendo um padrão de expressão regular. A segunda está informando a obrigatoriedade de preenchimento do campo e definindo seu tipo como sendo de email.

Imagine que vamos definir as pseudo-classes CSS **:valid** e **:invalid** podem ser usadas:

```
input:valid {  
    border: 2px solid green;  
}  
  
input:invalid {  
    border: 2px solid red;  
}
```

Veja no quadro abaixo o exemplo completo:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemplo de validação embutida do HTML5</title>
  <style>
    input:valid {
      border: 2px solid green;
    }

    input:invalid {
      border: 2px solid red;
    }
  </style>
</head>
<body>
  <form id="myForm" action="">
    <label for="userName">Nome:</label>
    <input type="text" id="userName" name="userName" minlength="4" maxlength="15" pattern="[A-Za-z]+" required>

    <label for="userEmail">E-mail</label>
    <input type="email" id="userEmail" name="userEmail" placeholder="Digite seu email" required>

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

Veja o resultado na tela do navegador, conforme figura abaixo:

Nome: E-mail:

Figura 1 - Exemplo de validação do formulário mostrando em funcionamento as pseudo-classes CSS :valid e :invalid

Descrição da imagem: O formulário contém dois campos de entrada: um para o nome e outro para o e-mail, seguidos por um botão "Enviar". Ao preencher o formulário, o navegador automaticamente verifica se os valores inseridos nos campos atendem aos critérios especificados: 1) O campo "Nome" deve ter entre 4 e 15 caracteres e conter apenas letras; 2) O campo "E-mail" deve ser um endereço de e-mail válido; 3) O usuário é notificado visualmente através das bordas dos

campos (verde para válido e vermelho para inválido), mas um usuário cego precisaria de um leitor de tela ou outra tecnologia assistiva para saber se os campos são válidos ou inválidos, pois essas mensagens visuais não seriam percebidas.

Os atributos de validação embutidos do HTML5 fornecem um mecanismo eficiente para validar a entrada do usuário, economizando tempo de desenvolvimento e fornecendo uma experiência de usuário mais consistente.

Veremos abaixo mais informações sobre os atributos de validação embutidos do HTML5.

Campos obrigatórios

O atributo de validação HTML5 mais simples de usar é o **required**. Se você precisa que um campo seja obrigatório, basta adicionar esse atributo ao elemento. Quando o atributo **required** está presente, o navegador impede que o formulário seja enviado se o campo correspondente estiver vazio, exibindo uma mensagem de erro ao usuário e marcando a entrada como inválida. Além das mensagens de erro, os campos marcados com **required** serão estilizados de acordo com as pseudo-classes **:invalid** e **:valid**, permitindo personalizar visualmente a aparência dos campos vazios ou preenchidos.

O atributo **required** pode ser usado em vários tipos de campos (**text**, **email**, **number**, **password**, etc.), bem como em menus suspensos (**select**), caixas de seleção (**checkbox**), e botões de opção (**radio**).

Considere o seguinte exemplo, conforme quadro abaixo:

```
<form id="myForm" action="">
  <label for="userName">Nome:</label>
  <input type="text" id="userName" name="userName" required>

  <label for="userAge">Idade:</label>
  <input type="number" id="userAge" name="userAge" required>

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

Neste exemplo, os campos **userName** e **userAge** devem ser preenchidos. Caso o usuário tente enviar o formulário sem inserir um valor em algum desses campos, o navegador exibirá uma mensagem de erro e bloqueará a submissão do formulário.

Veja o resultado no navegador, conforme figura abaixo:

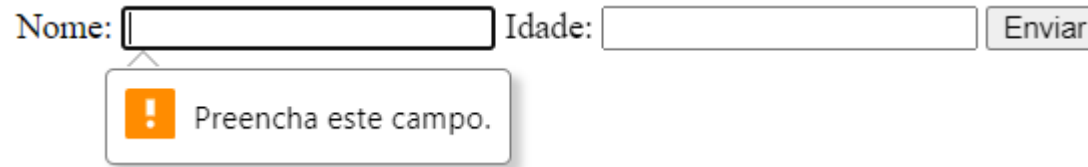


Figura 2 - Exemplo de uso da palavra-chave "required"

Descrição da imagem: No navegador, observamos um formulário simples com dois campos de entrada e um botão de envio. O primeiro campo é para digitar o nome. O segundo campo é para inserir a idade como um número. Um botão de envio para enviar os dados do formulário. Ao tentar enviar o formulário sem preencher esses campos, o navegador notificará que eles são obrigatórios.

Controlando tamanhos e intervalos de formulários HTML5

Os atributos **minlength** e **maxlength** no HTML5 fornecem uma forma nativa e eficaz de validar a entrada de dados de texto, controlando o comprimento mínimo e máximo de caracteres que o usuário pode inserir. Eles são úteis para garantir que as entradas de texto forneçam dados com a qualidade e o tamanho necessários, ajudando a prevenir erros de validação. Veja o exemplo no quadro abaixo:

```
<form id="myForm" action="">
  <label for="username">Nome de usuário (4-15 caracteres):</label>
  <input type="text" id="username" name="username" minlength="4" maxlength="15" required>

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

Para campos numéricos (**<input type="number">**), os atributos **min** e **max** fornecem restrições de validação que garantem que os valores estejam dentro de um intervalo específico. De maneira que:

- **min:** Define o valor mínimo permitido no campo numérico.
- **max:** Define o valor máximo permitido no campo numérico.

Ainda, o atributo **step** define o incremento dos valores permitidos para os tipos numéricos ou de data/hora. Se o valor fornecido não estiver alinhado com o **step** definido, o campo será considerado inválido.

Considere o seguinte exemplo, conforme quadro abaixo:

```
<form id="myForm" action="">
  <label for="even">Somente números naturais pares até 100:</label>
  <input type="number" id="even" name="even" min="0" max="100" step="2" required>

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

Veja o resultado no navegador, conforme figura abaixo:

Somente números naturais pares até 100:



Insira um valor válido. Os dois valores válidos mais próximos são 2 e 4.

Figura 3 - Exemplo de como controlar os tamanhos e intervalos de formulários HTML5

Descrição da imagem: No navegador, observamos um formulário com um campo de entrada numérico e um botão de envio. O campo numérico é rotulado como "Somente números naturais pares até 100". Se um usuário tentar enviar o formulário sem preencher o campo ou com um número que não seja par ou que esteja fora do intervalo de 0 a 100, o navegador notificará sobre a necessidade de corrigir a entrada.

Tipos de dados

O atributo **type** no HTML5 é usado para definir o tipo de dados que um campo **<input>** deve aceitar. Ele não apenas fornece um contexto claro ao usuário sobre que tipo de informação deve ser inserida, mas também ajuda na validação e apresentação de dados específicos. Além disso, permite que os navegadores forneçam interfaces específicas e adequadas ao tipo de entrada, como teclados personalizados para campos numéricos ou seletores de data.

Veja um exemplo, conforme quadro abaixo:

```
<form id="myForm" action="">
  <input type="text" name="username" placeholder="Digite seu nome">
  <input type="number" name="age" placeholder="Digite sua idade">
  <input type="email" name="userEmail" placeholder="Digite seu email">
  <input type="url" name="website" placeholder="https://example.com">
  <input type="password" name="password" placeholder="Digite sua senha">
  <input type="range" name="volume" min="0" max="100" step="1">
  <button type="submit">Enviar</button>
</form>
```

Expressão regular

O atributo **pattern** do HTML5 é um recurso poderoso para validação de formulários, permitindo que você especifique padrões personalizados para campos de entrada. Ele funciona principalmente com campos de texto, como **text**, **search**, **url**, **tel**, e **email**. Ele espera uma expressão regular (**regex**) como valor, que é usada para validar a entrada do usuário contra padrões predefinidos.

Uma expressão regular é um padrão que identifica combinações de caracteres em uma string de texto. Elas são ideais para validar formulários, mas também têm muitos outros usos em JavaScript, como encontrar e substituir texto.

Sua sintaxe básica é a seguinte, conforme quadro abaixo:

```
<input type="text" pattern="regex" title="message">
```

Onde:

- **pattern**: A regex que define o padrão de validação.
- **title**: Mensagem opcional que é mostrada ao usuário se a entrada for inválida.

Um padrão de expressão é composto por um conjunto de caracteres simples, como **/abc/**, ou uma combinação de caracteres simples e especiais, como **/ab*c/**. Neste contexto, os padrões simples são construídos com os caracteres que você deseja encontrar na sequência específica. Por exemplo, o padrão **/abc/** encontrará combinações de caracteres somente quando os caracteres **'abc'** aparecerem juntos e na ordem correta. Esse padrão será encontrado com sucesso nas strings "Olá, você conhece o **abc**?" e "Os mais recentes aviões evoluíram do **slabcraft**". Em ambos os casos, a correspondência estará no subconjunto **'abc'**.

Quando precisamos buscar padrões que vão além de uma correspondência direta, como encontrar múltiplas ocorrências de um caractere específico ou identificar espaços em branco, é necessário adicionar caracteres especiais ao padrão. Se você quiser encontrar a letra 'a' seguida por zero ou mais ocorrências da letra 'b', e depois 'c', a expressão regular correta seria `/ab*c/`. Veja que o asterisco (*) é um quantificador que indica "zero ou mais ocorrências" do item anterior, neste caso, a letra 'b'.

Na tabela abaixo estão alguns dos caracteres especiais mais importantes usados em expressões regulares, com suas respectivas descrições.

Tabela 1: Caracteres especiais usados em expressões regulares

Caractere	Descrição
.	Corresponde a qualquer caractere, exceto quebras de linha.
*	Zero ou mais ocorrências do item anterior.
+	Uma ou mais ocorrências do item anterior.
?	Zero ou uma ocorrência do item anterior (tornando-o opcional).
{n}	O caractere que precede o sinal pode ocorrerá n vezes
[abc]	Corresponde a qualquer caractere dentro dos colchetes.
[a-z]	Associa um dos caracteres de 'a' até 'z' em caixa baixa, lembre-se que expressões regulares são case-sensitive
\d	Qualquer dígito (0-9).
\w	Qualquer caractere alfanumérico (a-z, A-Z, 0-9, _).
\s	Qualquer espaço em branco (espaços, tabs, quebras de linha).

^	Corresponde ao início do texto
\$	Corresponde ao final do texto.

Veja um exemplo no quadro abaixo:

```
<form id="myForm" action="">
  <label for="cep">CEP (Formato: 12345-678):</label>
  <input type="text" id="cep" name="cep" pattern="\d{5}-\d{3}" title="Formato: 12345-678" required>
  <button type="submit">Enviar</button>
</form>
```

◀ 6.2 Validação do lado do cliente

Seguir para...

6.2.2 Validando formulários com JavaScript ►

[Baixar o aplicativo móvel.](#)