

Trabalhando formulários com JavaScript e validações

1. Introdução

Embora o HTML5 ofereça recursos nativos de validação de formulários, o JavaScript permite um controle muito mais refinado sobre esse processo, possibilitando validações personalizadas, mensagens de erro específicas e verificações que dependem de regras de negócio particulares de cada aplicação. Combinar os elementos de formulário do HTML com a lógica programável do JavaScript é uma das tarefas mais comuns no desenvolvimento web, sendo essencial para garantir que os dados enviados a um servidor estejam corretos e completos.

2. Acessando elementos do formulário

O primeiro passo para manipular um formulário com JavaScript é acessar seus elementos por meio do DOM (Document Object Model), geralmente utilizando métodos como "getElementById" ou "querySelector":

```
<form id="formCadastro">
  <input type="text" id="nome" name="nome">
  <button type="submit">Enviar</button>
</form>

<script>
  const formulario = document.getElementById("formCadastro");
  const campoNome = document.getElementById("nome");
</script>
```

3. Interceptando o envio do formulário

Para realizar validações antes do envio dos dados ao servidor, é comum "escutar" o evento "submit" do formulário e utilizar o método "preventDefault()" para impedir o envio caso alguma validação falhe:

```
formulario.addEventListener("submit", function (event) {
  if (campoNome.value.trim() === "") {
    event.preventDefault();
    alert("O campo nome é obrigatório.");
  }
});
```

Nesse exemplo, caso o campo "nome" esteja vazio (verificado após a remoção de espaços em branco com ".trim()"), o método "preventDefault()" impede que o formulário seja enviado, exibindo um alerta ao usuário.

4. Técnicas comuns de validação

- **Verificação de campos vazios** — confere se o valor de um campo, após remover espaços em branco, não é uma string vazia.
- **Validação de formato com expressões regulares (regex)** — utiliza padrões para verificar se um valor segue um formato esperado, como um endereço de e-mail ou um número de telefone.
- **Comparação entre campos** — verifica se dois campos coincidem, como na confirmação de senha em um cadastro.
- **Validação de intervalo numérico** — confere se um valor numérico está dentro de um intervalo aceitável, como idade ou quantidade de itens.

4.1 Exemplo com expressão regular

```
const regexEmail = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;  
  
if (!regexEmail.test(campoEmail.value)) {  
  alert("Digite um e-mail válido.");  
}
```

O método ".test()" verifica se o valor do campo corresponde ao padrão definido pela expressão regular, retornando "true" ou "false" conforme o resultado da comparação.

5. Fornecendo feedback visual ao usuário

Além de impedir o envio de dados inválidos, é uma boa prática fornecer retorno visual imediato ao usuário, geralmente adicionando ou removendo classes CSS que destacam campos com erro:

```
if (campoEmail.value.trim() === "") {  
  campoEmail.classList.add("campo-invalido");  
} else {  
  campoEmail.classList.remove("campo-invalido");  
}
```

6. Exemplos de aplicação

A validação de formulários com JavaScript é aplicada em uma ampla variedade de contextos no desenvolvimento web:

- **Cadastro de novos usuários** — verifica se a senha e a confirmação de senha coincidem, além de exigir um número mínimo de caracteres antes de permitir o envio do formulário.
- **Formulários de checkout em e-commerce** — valida se o número do cartão de crédito, a data de validade e o CEP de entrega estão preenchidos em formatos válidos antes de prosseguir com o pagamento.

- **Formulários de contato** — verifica se o campo de e-mail segue um formato válido e se a mensagem possui um tamanho mínimo, evitando o envio de formulários incompletos.
- **Sistemas de agendamento on-line** — valida se a data e o horário selecionados pelo usuário são válidos e estão dentro do período de funcionamento disponível.
- **Formulários de pesquisa de satisfação** — garante que ao menos uma opção tenha sido selecionada em perguntas de múltipla escolha antes de permitir o avanço para a próxima etapa.
- **Painéis administrativos com edição de dados** — valida em tempo real, conforme o usuário digita, se os valores inseridos em campos numéricos (como preço ou estoque) estão dentro de limites aceitáveis para o sistema.

7. Considerações finais

A validação de formulários com JavaScript complementa os recursos nativos do HTML5, permitindo verificações mais elaboradas e um retorno mais imediato e personalizado ao usuário. É importante destacar, no entanto, que a validação no lado do cliente (por meio do JavaScript no navegador) não substitui a validação no lado do servidor, já que esta última é indispensável para garantir a segurança e a integridade dos dados, mesmo em casos em que o JavaScript do cliente seja desabilitado ou manipulado.