

IOS – Instituto de
Oportunidade Social

JS 14 - Manipulando o DOM Parte 4



- Compreender o uso do DOM (Document Object Model);
- Aplicar código JavaScript para trabalhar com formulários;
- Conhecer os métodos `addEventListener` e `preventDefault`;
- Validações com `RegExp`.

IOS – Instituto de
Oportunidade Social

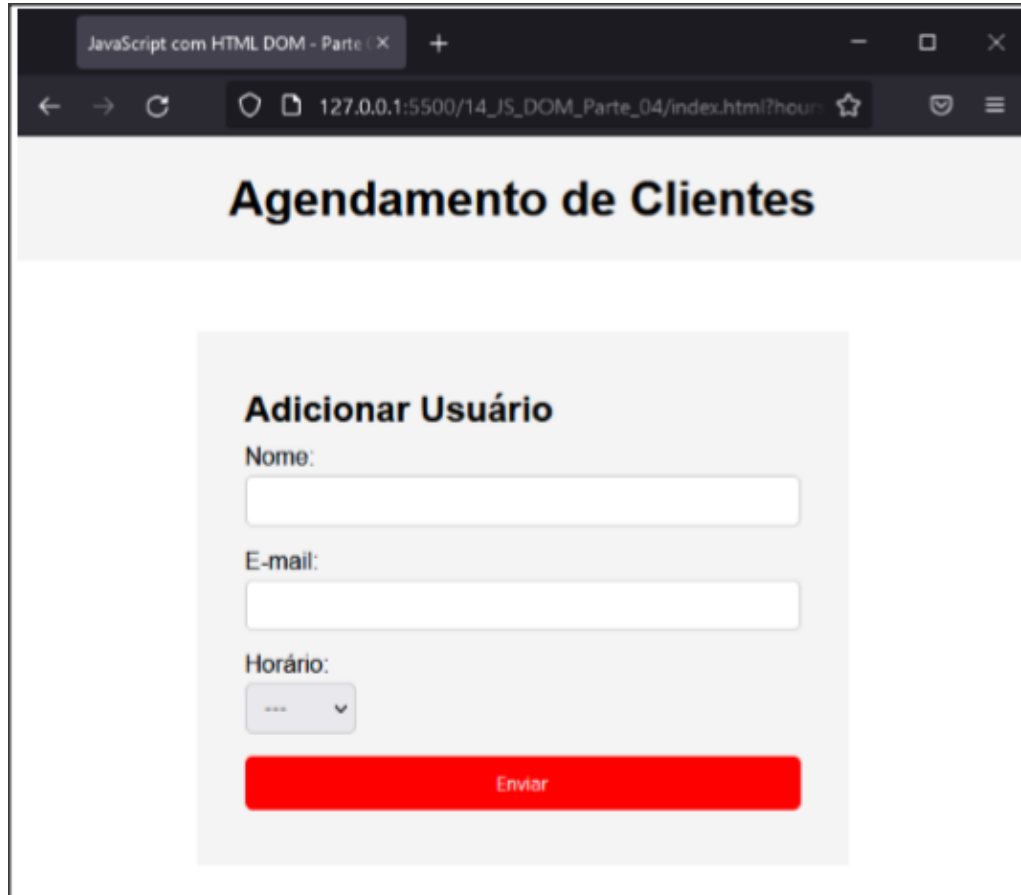
Utilizando JS em Formulário



A aplicação que vamos trabalhar é um **formulário de agendamento de clientes**. O agendamento será mostrado apenas na **interface do usuário**, não vamos salvar em **nenhum servidor de backend**, portanto **se a página recarregar os dados serão perdidos**. Então, a nossa agenda de clientes funcionará da seguinte forma:

- Você irá preencher o **Nome** e o **E-mail**, observe que o campo está com o atributo do **HTML required**.
- Assim que os **campos** estiverem **corretamente preenchidos** e você clicar no **botão submete**, as informações serão exibidas na **lista** `` disponível no código HTML.

Formulário de agendamento de clientes:



JavaScript com HTML DOM - Parte 1 X +

← → ↻ 127.0.0.1:5500/14_JS_DOM_Parte_04/index.html?hour=...

Agendamento de Clientes

Adicionar Usuário

Nome:

E-mail:

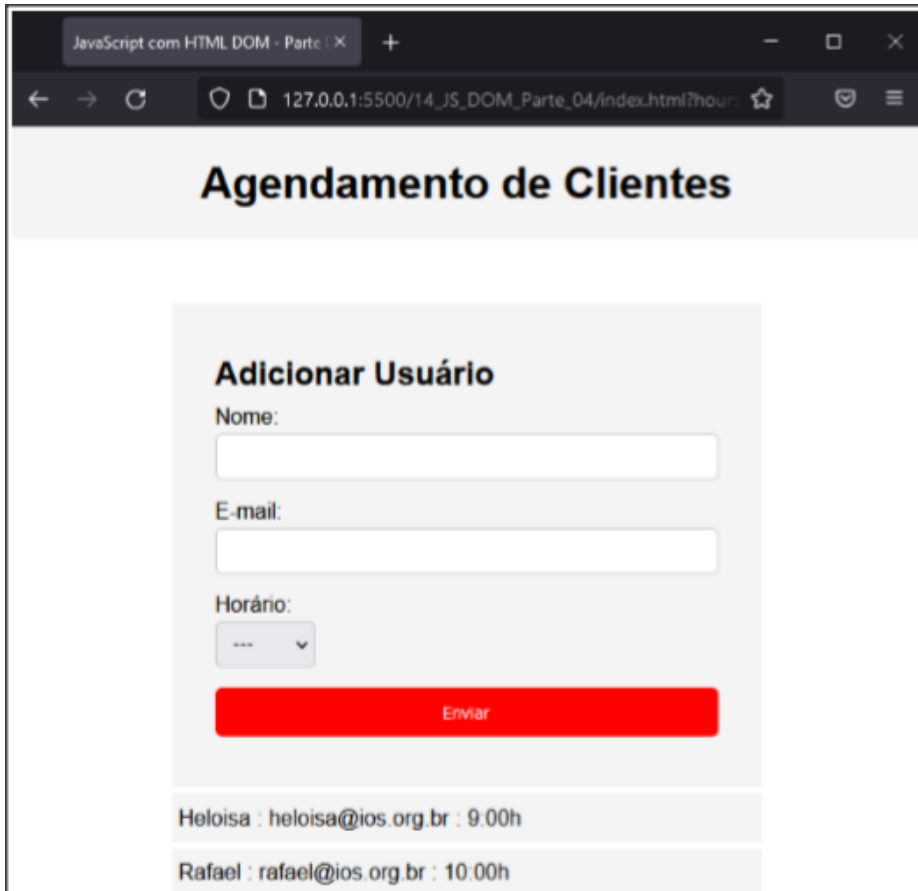
Horário:

...

Enviar

Utilizando JS em Formulário

Formulário com agendamentos:



JavaScript com HTML DOM - Parte : X

127.0.0.1:5500/14_JS_DOM_Parte_04/index.html?hour

Agendamento de Clientes

Adicionar Usuário

Nome:

E-mail:

Horário:

Enviar

Heloisa : heloisa@ios.org.br : 9:00h

Rafael : rafael@ios.org.br : 10:00h

Método addEventListener

O método **addEventListener** “vigia” um evento de um elemento específico. Esse evento pode ser o clique de um botão, a submissão de um formulário, etc. No nosso exemplo, vamos utilizar esse método para “vigiar” o evento de submissão do formulário.

```
// Método addEventListener  
myForm.addEventListener('submit', onSubmit);
```

Demais Métodos e atributos:

O método **preventDefault()** evita que a página seja recarregada pela ação de submeter o formulário, pois essa é uma **ação padrão** do HTML quando estamos trabalhando com **formulários**.

A propriedade **classList.add** adiciona uma classe em um elemento

O método **appendChild()** adiciona um nó ao final da lista de elementos filhos de um elemento pai.

O método **.focus()** posiciona o cursor num determinado elemento

IOS – Instituto de
Oportunidade Social

Validando e-mail com RegEx



O **Regex** (regular expression) é uma **sequência de caracteres** que especifica um **padrão de busca**. Normalmente, esses padrões são usados por algoritmos de busca de strings para realizar operações de "**localizar**" ou de "**localizar e substituir**" e também para **validação de entrada de dados**. Você pode testar qualquer tipo de expressão regular no site <https://regex101.com/>.

Exemplo de uso do RegEx:

```
// Validando e-mail
emailInput.addEventListener('change', (e) => {
    let padrao = new RegExp(/.*@.*\..*/i);
    if (!padrao.test(emailInput.value)) {
        // alert('Por favor, insira um e-mail válido.');
```

msg_email.classList.add('error');

msg_email.innerHTML = 'Por favor, insira um e-mail válido.';

setTimeout(() => msg.remove(), 3000);

```
    }
});
```

Analizando o uso do RegEx:

A instrução vincula a execução de uma **Arrow Function** ao evento **change** do **campo de email**. Quando você terminar de digitar e sai do campo, a função de verificação do e-mail é executada. A expressão testada é `/.*@.*\..*/i`, que verifica se **tem um símbolo de arroba (@)** e um **ponto final no e-mail**. Não é a melhor das validações de e-mail, mas funciona para o nosso caso.

Caso o e-mail esteja inválido de acordo com o teste lógico `!padrao.test(emailInput.value)`, a mensagem de erro irá aparecer.

IOS – Instituto de
Oportunidade Social

Vamos Praticar



Apostila de JS

04.JavaScript

Páginas 173 a 181

OBS: Acompanhar o passo a passo com o instrutor

IOS – Instituto de
Oportunidade Social

Para aprender mais



Para aprender mais



Procure sempre aprender e estudar mais. Seguem alguns links para você estudar e aprender mais:

addEventListener:

<https://developer.mozilla.org/pt-BR/docs/Web/API/EventTarget/addEventListener>

[https://www.w3schools.com/jsref/met element addeventlistener.asp](https://www.w3schools.com/jsref/met_element_addeventlistener.asp)

appendChild:

<https://developer.mozilla.org/pt-BR/docs/Web/API/Node/appendChild>

[https://www.w3schools.com/jsref/met node appendchild.asp](https://www.w3schools.com/jsref/met_node_appendchild.asp)

Para aprender mais



Regex:

<https://regex101.com/>

[https://en.wikipedia.org/wiki/Regular expression](https://en.wikipedia.org/wiki/Regular_expression)

Javascript e Formulários:

<https://www.youtube.com/watch?v=OR8ySydmqLQ>

<https://www.youtube.com/watch?v=evS7nVlbsw4>

<https://www.youtube.com/watch?v=qMxhzUh2gUc>

<https://www.youtube.com/watch?v=NoZOqtK6ecl>

IOS – Instituto de
Oportunidade Social

Exercícios



- Criar um form com 4 campos (nome, dataNasc, email e item) e 2 botões Incluir e Excluir. O botão **Incluir** deverá trabalhar com os campos **nome**, **dataNasc** e **email**, criando uma lista de inscritos (**maiores de 18 anos**) e e-mail validado por **Regex**, caso não seja maior ou e-mail inválido emitir a mensagem e não incluir no formulário. O botão **Excluir** trabalhará com o campo **item**, onde o código de Item informado em conjunto com a ação de click deverá remover o inscrito correspondente.

```
let data      = new Date();           // Obtém a data/hora atual
let dia       = data.getDate();       // 1-31
let mes       = data.getMonth();     // 0-11 (zero=janeiro)
let ano4      = data.getFullYear();  // 4 dígitos
let str_data  = dia + '/' + (mes+1) + '/' + ano4; // Formata a data e a hora (note o mês + 1)
```