DOM PARA INICIANTES

Eventos

addEventListener

Adiciona uma função ao elemento, esta chamada de `callback`, que será ativada assim que certo `evento` ocorrer neste elemento.

```
const img = document.querySelector('img');

// elemento.addEventListener(event, callback, options)
img.addEventListener('click', function() {
  console.log('Clicou');
})
```

> *O terceiro parâmetro é opcional.*

Callback

É boa prática separar a função de callback do addEventListener, ou seja, declarar uma função ao invés de passar diretamente uma função anônima

```
const img = document.querySelector('img');
function callback() {
  console.log('Clicou');
}

img.addEventListener('click', callback); // 🚀
img.addEventListener('click', callback()); // undefined
img.addEventListener('click', function() {
  console.log('Clicou');
})
img.addEventListener('click', () => {
  console.log('Clicou');
})
```

Event

O primeiro parâmetro do callback é referente ao evento que ocorreu.

```
const img = document.querySelector('img');

function callback(event) {
  console.log(event);
}

img.addEventListener('click', callback);
```

> *Geralmente utilizam `e` como nome do parâmetro*

Propriedades do Event

```
const animaisLista = document.querySelector('.animais-lista');

function executarCallback(event) {
  const currentTarget = event.currentTarget; // this
  const target = event.target; // onde o clique ocorreu
  const type = event.type; // tipo de evento
  const path = event.path;
  console.log(currentTarget, target, type, path);
}

animaisLista.addEventListener('click', executarCallback);
```

event.preventDefault()

Previne o comportamento padrão do evento no browser. No caso de um link externo, por exemplo, irá previnir que o link seja ativado.

```
const linkExterno = document.querySelector('a[href^="http"]');

function clickNoLink(event) {
  event.preventDefault();
  console.log(event.currentTarget.href);
}

linkExterno.addEventListener('click', clickNoLink);
```

this

A palavra chave `this` é uma palavra especial de JavaScript, que pode fazer referência a diferentes objetos dependendo do contexto. No caso de eventos, ela fará referência ao elemento em que addEventListener foi adicionado.

```
const img = document.querySelector('img');

function callback(event) {
  console.log(this); // retorna a imagem
  console.log(this.getAttribute('src'));
}

img.addEventListener('click', callback);
```

> *Geralmente igual ao event.currentTarget*

Diferentes Eventos

Existem diversos eventos como `click`, `scroll`, `resize`, `keydown`, `keyup`, `mouseenter` e mais. Eventos podem ser adicionados a diferentes elementos, como o `window` e `document` também.

```
const h1 = document.querySelector('h1');

function callback(event) {
  console.log(event.type, event);
}

h1.addEventListener('click', callback);
h1.addEventListener('mouseenter', callback);
window.addEventListener('scroll', callback);
window.addEventListener('resize', callback);
window.addEventListener('keydown', callback);
```

> https://developer.mozilla.org/en-US/docs/Web/Events

Keyboard

Você pode adicionar atalhos para facilitar a navegação no seu site, através de eventos do `keyboard`.

```
function handleKeyboard(event) {
  if(event.key === 'a')
    document.body.classList.toggle('azul');
  else if(event.key === 'v')
    document.body.classList.toggle('vermelho');
}

window.addEventListener('keydown', callback);
```

forEach e Eventos

O método `addEventListener` é adicionado à um único elemento, então é necessário um loop entre elementos de uma lista, para adicionarmos à cada um deles.

```
const imgs = document.querySelectorAll('img');

function imgSrc(event) {
  const src = event.currentTarget.getAttribute('src');
  console.log(src);
}

imgs.forEach((img) => {
  img.addEventListener('click', imgSrc);
});
```

Exercício

```
// Quando o usuário clicar nos links internos do site,
// adicione a classe ativo ao item clicado e remova dos
// demais itens caso eles possuam a mesma. Previna
// o comportamento padrão desses links


// Selecione todos os elementos do site começando a partir do
body,
// ao clique mostre exatamente quais elementos estão sendo
clicados


// Utilizando o código anterior, ao invés de mostrar no
console,
// remova o elemento que está sendo clicado, o método remove()
remove um elemento


// Se o usuário clicar na tecla (t), aumente todo o texto do
site.
```