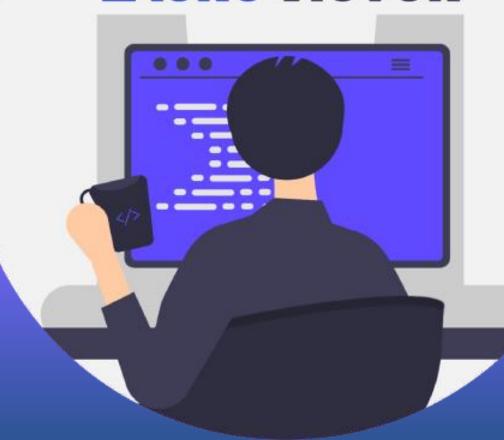
LAURO VICTOR





Especificando eventos

- Existem três maneiras de especificar eventos:
- 1. Pelo HTML
- 2. Pelo JavaScript
- 3. Pelos **event listeners**



Especificando eventos pelo HTML

• O melhor de especificar eventos dessa maneira é que é muito fácil ler esse código e prever o que ele fará. Assim que você clicar no parágrafo, a função **magic()** será acionada. Existem algumas desvantagens também: você só pode especificar um evento dessa maneira e também não pode alterar o evento dinamicamente.

Click here for magic!



Especificando eventos pelo JavaScript

Aqui está a primeira maneira de fazer isso usando JavaScript

```
document.getElementById("unique").onclick = function() {
          magic();
};
```

 Estamos pegando a propriedade que representa o evento selecionado e atribuindo nossa função a ele. Então, neste caso, estamos selecionando o p mostrado na seção anterior por seu valor de atributo único, pegando a propriedade onclick e atribuindo a função magic() a ele envolvendo-o em uma função anônima.

Event Listeners

- O último método é usar o método **addEventListener()** para adicionar um evento a um elemento. Com isso, podemos especificar várias funções para um mesmo evento, por exemplo, quando um elemento é clicado.
- O que chama a atenção para ambos os métodos que vimos—usando eventos HTML e atribuindo propriedades—é que o evento recebe o prefixo on. Por exemplo, onclick, onload, onfocus, onblur, onchange, etc. Este não é o caso quando usamos o método addEventListener(), onde especificamos o tipo de evento dentro do ouvinte de eventos sem o prefixo on.



Event Listeners

document.getElementById("unique").addEventListener("click", magic);

 Observe que estamos deixando de fora os parênteses por trás da função mágica aqui. Não podemos enviar parâmetros como este. Se isso é algo que você precisa fazer, você teria que envolver a funcionalidade em uma função anônima, assim:



Evento OnLoad

 O evento onload é acionado depois que um determinado elemento é carregado. Isso pode ser útil por vários motivos. Por exemplo, se você quiser selecionar um elemento usando getElementByld, terá que ter certeza de que esse elemento já está carregado no DOM. Esse evento é mais comumente usado no objeto window, mas pode ser usado em qualquer elemento.

```
window.onload = function() {
    // Fazer algo após a página ser completamente carregada
}
```



Evento OnLoad

- Você também pode usar o método addEventListener() em qualquer elemento para manipular qualquer evento. E também pode ser usado para o evento em que todo o conteúdo do DOM seja carregado. Existe um evento embutido especial para isso: DOMContentLoaded().
- Este evento pode ser usado para lidar com o evento de carregamento do DOM, que será acionado imediatamente após o DOM da página ser construído quando o evento for definido.

```
document.addEventListener("DOMContentLoaded", (e) => {
   console.log(e);
});
```



Propriedade target de um evento

• Sempre que um evento é acionado, uma variável de evento fica disponível. Ele tem muitas propriedades, e você pode conferir usando este comando na função que é acionada para o evento.

console.log(event);

 Isso mostrará muitas propriedades. Uma das propriedades mais interessantes por enquanto é a propriedade target. O target é o elemento HTML que disparou o evento. Assim, podemos usá-lo para obter informações de uma página da web. Vejamos um exemplo simples.



Propriedade target de um evento



Propriedade target de um evento

• Sempre que um evento é acionado, uma variável de evento fica disponível. Ele tem muitas propriedades, e você pode conferir usando este comando na função que é acionada para o evento.

console.log(event);

 Nesse caso, event.target é o elemento de botão. No console, o elemento botão e todas as suas propriedades serão registradas, incluindo possíveis irmãos e pais.

let pai = event.target.parentElement;



Obrigado!



