Quando eu clico nos links internos nos menus eles dão um pulo e isso pode parecer pro cliente que o link foi pra uma página diferente e com o scroll suave ele fica com a impressão de que está na mesma página por isso que ele é importante.

Como posso fazer isso?

1° Vou selecionar a tag nav pertecente ao menu e adicionar a classe js-menu:

```
<nav class="menu js-menu">
        <ul>
            <li><a href="#animais">Animais</a></li>
            <li><a href="#faq">Faq</a></li>
            <li><a href="#contato">Contato</a></li>
        </ul>
    </nav>
```

JavaScript:

```
const linksInternos = document.querySelectorAll('.js-menu
a[href^="#"]');
```

Output: NodeList(3) [a, a, a]

Porque NodeList?

Porque objetos NodeList são coleções de nodos semelhantes aos objetos retornados pelos métodos Node.childNodes e document.querySelectorAll().

Qual a diferença entre NodeList e HTMLCollection?

O DOM (Document Object Model) contém inúmeros objetos, como window que representa uma janela do navegador, e documentque representa o conteúdo da página. E dentro desses objetos, podem conter listas de objetos, as NodeList.

De acordo com a documentação MDN objetos NodeList são coleções de nodos semelhantes aos objetos retornados pelos métodos Node.childNodes e document.querySelectorAll().

Em outras palavras, são coleções de objetos. Então, o DOM é formado de objetos, que podem conter NodeList de outros objetos.

Acessando a documentação linkada acima você consegue ver exemplos muito interessantes para entender melhor.

E quanto ao HTMLCollection, ele representa uma coleção genérica (objeto array) de elementos (na ordem em que aparecem no documento) e oferece métodos e propriedades para selecioná-los da lista.

Também indico a leitura da documentação MDN.

Em resumo, o HTMLColection é uma coleção de elementos HTML (div, p, a, img, etc...), e o NodeList é uma coleção de nós, como no caso do curso. E por isso é utilizado.

```
const linksInternos = document.querySelectorAll('.js-menu
a[href^="#"]');

linksInternos.forEach((link) => {
  link.addEventListener('click', scrollToSection);
});
```

Para cada link selecionado essa função vai fazer o scroll pra sessão do link.

A relação entre o link e o menu é dado pelo id e pelo href.

```
function scrollToSection(event) {
  event.preventDefault();
}
```

event.preventDefault vai prevenir aquele evento padrão.

Agora eu vou pegar o href do link que eu clicar:

```
const href = event.currentTarget.getAttribute('href');
console.log(href);
```

currentTarget.getAtrribute(): Pega o atributo href.
currentTarget.href = Pega o link inteiro.

Output: #contato, #animais, #faq.

```
function scrollToSection(event) {
  event.preventDefault();
  const href = event.currentTarget.getAttribute('href');
  const section = document.querySelector(href);
  console.log(section);
}
```

Pra eu fazer o scroll do link até a seção dele eu posso usar os seguintes métodos:

scrollTo();

Ele leva dois argumentos o eixo X e o eixo Y. O X seria se eu mexesse ele na horizontal e como eu quero na vertical então eu quero o eixo Y.

```
window.scrollTo(0, 1000);
```

Ainda não ocorreu o scroll suave porque ele está com um valor fixo.

Então uma forma que eu posso fazer é pegando o topo da sessão:

```
const topo = section.offsetTop;
window.scrollTo(0, topo);
```

Ainda é o formato padrão. No scrollTo há uma novidade chamada options, porém ela não é suportada em todos os navegadores ainda.

Então ao invés de passar o eixo X eu posso passar opções e passo como um objeto como argumento:

```
window.scrollTo({
    top: topo,
    behavior: 'smooth',
});
```

Agora sim eu criei o scroll suave com a propriedade **behavior** sem ela continua o formato padrão.

1° Forma:

```
const topo = section.offsetTop;

  window.scrollTo({
    top: topo,
    behavior: 'smooth',
  });
```

Porém, há uma outra forma de se fazer o scroll suave. Existe o scrollIntoView ele está relacionado ao elemento em si então eu posso falar com a section e chamar o scrollIntoView, ou seja, deu scroll bota ela na visão pra mim:

```
section.scrollIntoView(); // Comportamento padrão
```

Nele também tem como passar opções:

```
section.scrollIntoView({
    behavior: 'smooth',
});
```

Ele tem um comportamento diferente porque ele está indo até o meio da sessão ele não está alinhando certinho no topo então eu devo alinhar desde do começo:

```
section.scrollIntoView({
    behavior: 'smooth',
    block: 'start',
});
```

Eu não precisei mais ver qual a distância entre a section e o topo porque a section leva o elemento e eu trago esse elemento pra frente.

Função completa:

```
function initScrollSuave() {
  const linksInternos = document.querySelectorAll('.js-menu
a[href^="#"]');

  function scrollToSection(event) {
  event.preventDefault();
  const href = event.currentTarget.getAttribute('href');
  const section = document.querySelector(href);

  section.scrollIntoView({
    behavior: 'smooth',
    block: 'start',
  });
}

linksInternos.forEach((link) => {
  link.addEventListener('click', scrollToSection);
});
}
initScrollSuave();
```

Esse método só é suportado no Chrome no Firefox. Antigamente, se usava muito o JQuery que fosse consistente entre navegadores antigos e mais pra frente eu vou criar um plugin pra transformar esse meu código em código antigo pra que ser

suportado pra todos os navegadores antigos também e eu possa escrever ele de forma moderna, mas que ele continue funcionando ainda em navegadores antigos.