02

Dialog acessível

Transcrição

Se usarmos "Enter" na parte "Destaques" do site, o NVDA não avisa o que está acontecendo, apesar de ter aparecido um modal com "News sem spam", para assinatura da newsletter.

> Não existe exatamente uma convenção em torno disto, fiz até uma pesquisa rápida no Twitter, e a maioria das pessoas disse que esta janela é um modal, mesmo. Digamos que o termo mais "ortodoxo" seria *dialog*, tanto que no HTML 5.2 criaram um elemento com mesmo nome, mas isto fica para outra ocasião.

Como o leitor de telas saberá que um pop up foi aberto? Após teclarmos "Tab", selecionaremos o campo "Digite seu e-mail...", porém quem utiliza o leitor de telas não terá contexto e muito provavelmente se sentirá perdido.

Além disso, os elementos que se encontram atrás deste modal ainda são focáveis, o que acaba não sendo interessante, sendo necessário retirar isso. O que normalmente ocorre é que nós, front-enders, às vezes acabamos nos esquecendo da acessibilidade, e precisamos tomar cuidado.

Precisaremos melhorar o botão de fechar o modal também, já que se clicarmos em qualquer ponto além do pop up, ou apertarmos "Esc", o esperado é que a janela sumisse, ou fechasse, mas não é isto que acontece.

Feliz ou infelizmente, chegamos a este ponto do curso em que precisaremos ver um pouco de JavaScript, então, se você não tiver conhecimento disto ou de lógica de programação, recomenda-se fazer os cursos aqui na Alura.

Voltaremos ao código para fazer o botão funcionar via JavaScript, que terá um `display: none` em vez do `display: block`. Feito isso, salvaremos o arquivo e retornaremos ao navegador para ver se visualmente tudo está certinho.

Agora, via JavaScript, voltaremos para `display: block` usando o atalho "Ctrl + Shift + O", que é o *Quick open*, para abrirmos o arquivo `dialog.js`.

Nele, temos a função `fechandoDialog()`, a ser chamada quando clicamos no "x". O `dialogBody` é o corpo da `dialog`, como o próprio nome indica, e o `dialogOverlay` é basicamente a transparência que fica de fundo na janelinha.

Após as variáveis, é necessário alterar o `style` do `btnAbreDialog`. Poderíamos colocar uma classe, o que seria um pouco mais ortodoxo, mas não será o caso. O código ficará assim:

```
btnAbreDialog.style.display = 'block';
```

Aqui estamos colocando um CSS *inline* "forçado" via JavaScript. Salvaremos e voltaremos à nossa página, que continuará funcionando como esperado. Porém, utilizando-se o NVDA, nada é anunciado.

No arquivo HTML, a `dialog` é uma `div` "perdida", sem indicação de sua função ou objetivo. Por conta disto, acrescentaremos o `role="dialog"` para atribuirmos um papel a ela e melhorarmos a semântica.

Também não adianta abrirmos a *dialog* e indicarmos isto simplesmente. É uma *dialog* de que? O título é "News sem spam", então, outro ponto interessante seria que toda a `div` referente a `dialogNewsletter` tivesse um rótulo, que será justamente o `h2` com o título.

Pensando em acessibilidade, faremos com que algo rotule outro por meio de `aria-labelledby`. O código ficará da seguinte forma:

```
<div class="dialogNewsletter" id="dialogNewsletter" role="dialog" aria-labelledby="tituloDialog">
    <div class="dialogNewsletter-body">

        <h2 class="dialogNewsletter-titulo" id="tituloDialog">News sem spam</h2>
    <!--...-->
    </div>
</div>
```

Vamos testar?

> Uma dica é utilizar o atalho "Ctrl + R" caso você esteja lidando com o JavaScript no Brackets (http://brackets.io/), pois às vezes ele não carrega direito.

Voltando ao navegador e usando o NVDA, checaremos que ele ainda não indica que trata-se de uma *dialog*. O que será que está acontecendo? Apenas quando usamos o "Tab" é que o leitor diz "News sem spam diálogo".

Se quando damos "Enter" não se diz nada, mas quando usamos o "Tab" com foco do email ele começa a ser lido, o que poderíamos fazer?

E se colocássemos o foco no campo de preenchimento, no momento em que a pessoa clicar em "Receber destaques por email"?

Em `index.html`, portanto, há uma tag `<button>` com `id="abreDialog"`, justamente o que encontramos em `dialog.js`. Colocaremos nele um ouvinte de eventos para que, ao ser clicado, aconteça algo, no caso, se abra a janela.

Selecionando `dialogNewsletter--aberto` e usando o atalho "Ctrl + Shift + O", buscamos o que ele faz exatamente. Agora, precisaremos do campo em si. Pegamos um elemento no JavaScript por meio de `document.querySelector()`, e acrescentamos o foco com `focus()`:

```
//Quando abrir a dialog...
btnAbreDialog.addEventListener('click', function() {
    dialog.classList.add('dialogNewsletter--aberto');
    document.querySelector('.dialogNewsletter-campo').focus();
});
```

Lembrando que para o *label* saber que é deste campo, usaremos o `for="emailNewsletter"` e, no `<input>`, acrescentaremos `id="emailNewsletters"`. O `type="text"` faz com que se indique o uso do teclado mas, por ser um email, não seria legal se o teclado já viesse com o arroba (`@`) também?

Conseguimos fazer isto trocando `text` por `email`, do HTML5. Com isso, voltaremos à página, recarregaremos e testaremos o modal novamente. O NVDA desta vez lê

```
Receber destaques por email
botão
News sem spam diálogo
News sem spam diálogo
Digite seu e-mail... edição
```

```
possui autocompletar
em branco
```

que é o que queríamos!

Conseguimos acrescentar um `role="dialog"` para identificar que esta janelinha é um modal, e usamos o `aria-labelledby` para sincronizar e indicar que a janela é rotulada pelo `h2` "News sem spam".

A seguir continuaremos aplicando melhorias em relação à acessibilidade do modal, pois ainda não conseguimos fechá-lo clicando fora da janela ou simplesmente apertando "Esc".