

Aula 5

Navegação inclusiva

► Unidade

**Acessibilidade na web:
melhorando a experiência
do usuário**

O que vamos aprender?



Compreender a importância de um menu de acessibilidade para melhorar a usabilidade de páginas web.



Implementar a funcionalidade de exibir e ocultar opções de acessibilidade.



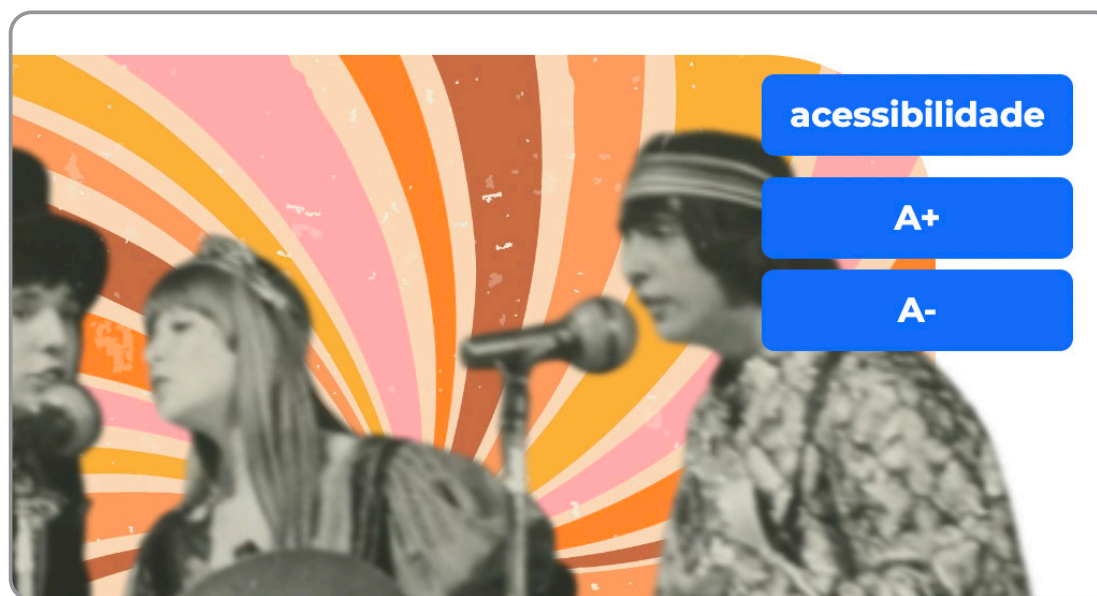
Utilizar manipulação de classes em JavaScript para controlar a exibição dinâmica de opções de acessibilidade, como ajuste de tamanho de fonte.



ACOMPANHE O VÍDEO DA AULA

Criando um menu de acessibilidade

Na última aula, deixamos o texto do nosso site acessível sem utilizar o recurso de zoom do navegador. Agora, precisamos deixar claro que esses botões são um recurso de acessibilidade. Nesta aula, aprenderemos a fazer um agrupamento de botões para criar um menu de acessibilidade.



Para iniciar, abriremos o arquivo *index.html* no VS Code e criaremos um novo botão logo antes da **div** que contém o **id="opcoes-acessibilidade"**. O código deve ficar da seguinte forma:

```
<nav class="container d-flex justify-content-between align-items-center" >
  
  <ul class="nav mt-5">
    <li class="nav-item"><a class="nav-link" href="#inicio">Início</a></li>
    <li class="nav-item"><a class="nav-link" href="#galeria">Galeria
  </a></li>
    <li class="nav-item"><a class="nav-link" href="#contato">Contato
  </a></li>
  </ul>
  <div id="acessibilidade">
    <button >acessibilidade</button>
    <div id="opcoes-acessibilidade">
      <button id="aumentar-fonte" class="btn btn-primary fw-bold">
A+</button>
      <button id="diminuir-fonte" class="btn btn-primary fw-bold">
A-</button>
    </div>
  </div>
</nav>
```

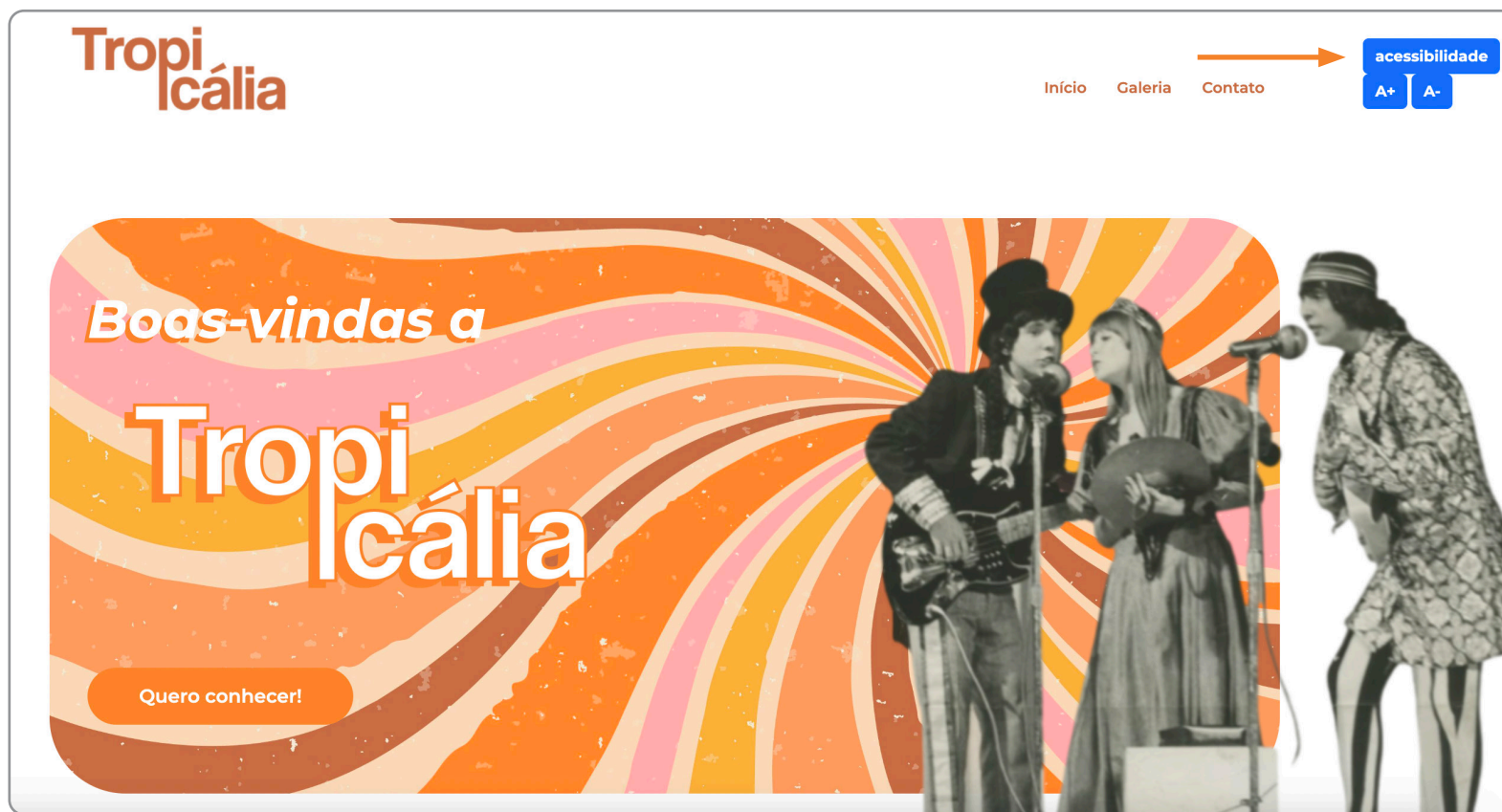
Em seguida, salve seu código, abra o arquivo *index.html* no navegador e verifique se o botão está sendo exibido. Ele deve aparecer junto dos outros botões de acessibilidade, no canto superior direito da tela. Observe:

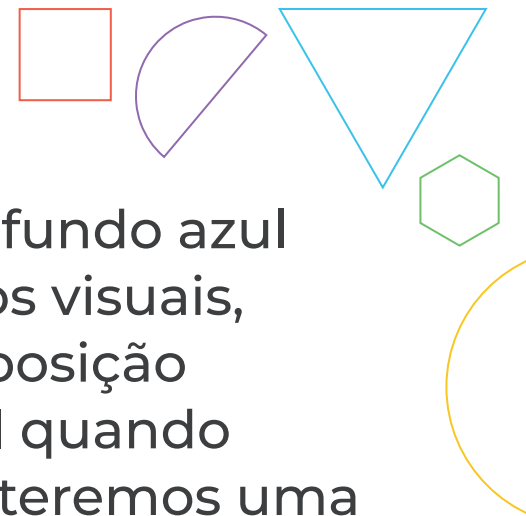


Para manter o padrão estético dos outros botões de acessibilidade que já fizemos, adicionaremos as mesmas classes de estilo ao novo botão. Para isso, adicionaremos o parâmetro de classes do CSS e as classes que o Bootstrap disponibiliza ao projeto. Portanto, o código do novo botão será **class="btn btn-primary fw-bold"**. Observe:

```
<nav class="container d-flex justify-content-between align-items-center" >
  
  <ul class="nav mt-5">
    <li class="nav-item"><a class="nav-link" href="#inicio">Início</a></li>
    <li class="nav-item"><a class="nav-link" href="#galeria">Galeria</a></li>
    <li class="nav-item"><a class="nav-link" href="#contato">Contato</a></li>
  </ul>
  <div id="acessibilidade">
    <button class="btn btn-primary fw-bold" >acessibilidade</button>
  <div id="opcoes-acessibilidade">
    <button id="aumentar-fonte" class="btn btn-primary fw-bold">A+</button>
    <button id="diminuir-fonte" class="btn btn-primary fw-bold">A-</button>
  </div>
</div>
</nav>
```


Agora que aplicamos a classe de estilo ao botão, salve novamente e atualize a página no seu navegador. O botão adicionado deve estar azul e com a fonte branca, como os demais botões de acessibilidade.

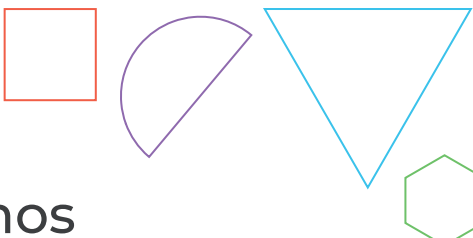




Agora, temos o botão de acessibilidade no menu. Ele tem o fundo azul e as letras brancas, assim como os botões A+ e A-. Em termos visuais, queremos que esse botão fique colado à direita da tela, na posição vertical. Ainda, seria interessante que volte para a horizontal quando clicarmos nele e, então, apresente os botões A+ e A-. Assim, teremos uma lista de opções. Vamos, então, trabalhar nessa estilização.

De volta ao arquivo *index.html* no VS Code, adicionaremos uma classe de estilo do CSS chamada **menu-acessibilidade** na **div** com **id="acessibilidade"**. Observe que, além disso, também adicionamos o parâmetro **rotacao-botao** à estilização do botão de acessibilidade para que ele possa rotacionar. O código da **div** deve ficar da seguinte forma:

```
<div id="acessibilidade" class="menu-acessibilidade">
  <button id="botao-acessibilidade" class="btn btn-primary fw-bold-
  rotacao-botao">acessibilidade</button>
  <div id="opcoes-acessibilidade">
    <button id="aumentar-fonte" class="btn btn-primary fw-bold">A+
  </button>
    <button id="diminuir-fonte" class="btn btn-primary fw-bold">A-
  </button>
  </div>
</div>
```

Apesar de já termos adicionado a classe ao botão, não veremos nenhuma alteração no site porque a classe ainda não existe no arquivo *style.css*. Então, abriremos esse arquivo e, no fim dele, criaremos uma classe chamada **.menu-acessibilidade { }**.

Dentro dessa classe, adicionaremos as propriedades **position: fixed;**, **top: 2rem;**, **height: 20px** e **z-index: 1000;**. Ao finalizar a implementação, o código da classe **.menu-acessibilidade** ficará da seguinte forma:

```
.menu-acessibilidade{  
  position: fixed;  
  top: 2rem;  
  right: 20px;  
  z-index: 1000;  
}
```

Salve seu trabalho e verifique como ficou seu site. Consegue perceber a diferença que essa classe causou na aparência dos botões?

O nosso site ficou da seguinte forma, depois de implementar a classe **menu-acessibilidade**:



Antes de implementar essa classe no arquivo CSS, os botões ficavam em uma única posição, sumindo ao rolarmos a página.

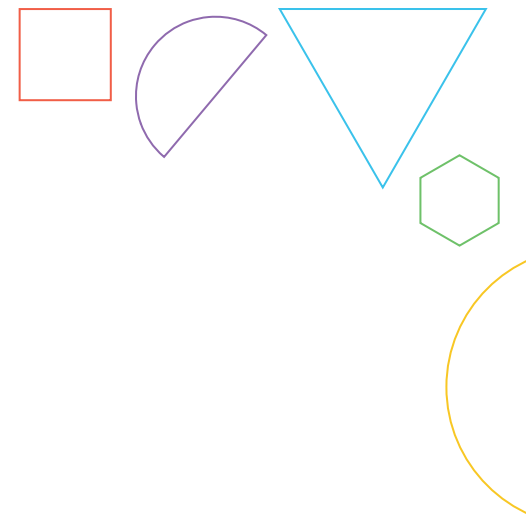
Com a classe implementada, os botões passam a ser fixos e aparecem sempre na mesma posição ao rolar para outras partes do site. Isso acontece por causa dos parâmetros **position: fixed** e **z-index: 1000**.

O **position: fixed** prende o item na tela sempre na mesma posição. Para deixar os botões sempre acima do restante do site, usamos o **z-index** com um número bem alto, garantindo que esteja na camada mais acima do site.



Agora, implementaremos duas novas classes que servirão para deixar o menu de acessibilidade na vertical e preso ao lado direito da tela. Primeiro, deixaremos os botões de aumentar e diminuir a fonte organizados em formato de lista e, em seguida, esconderemos os botões para que só sejam exibidos ao clicarmos no botão acessibilidade.

Para isso, no arquivo *style.css*, criaremos, logo abaixo do código da classe **.menu-acessibilidade**, a classe **.rotacao-botao { }** e adicionaremos as propriedades **transform: rotate(-90deg);** e **transform-origin: right center;** dentro dela.

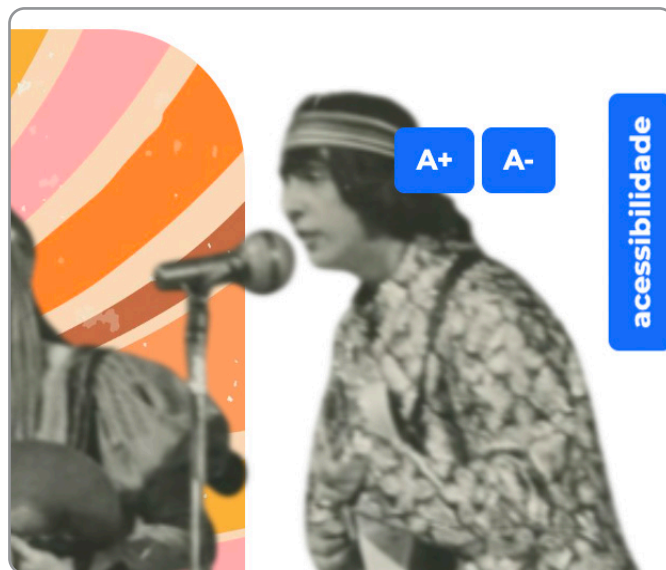


O código deve ficar da seguinte forma:

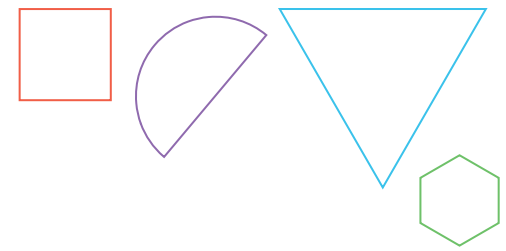
```
.menu-acessibilidade{  
  position: fixed;  
  top: 2rem;  
  right: 20px;  
  z-index: 1000;  
}  
  
.rotacao-botao{  
  transform: rotate(-90deg);  
  transform-origin: right center;  
}
```

Agora, precisamos dizer ao arquivo HTML que queremos usar essa nova classe.

Ao salvar o arquivo e verificar o navegador, teremos o seguinte resultado:



Agora, implementaremos algo que servirá para deixar os botões de aumentar e diminuir a fonte organizados em formato de lista, escondendo esses botões para que só sejam exibidos ao clicar no botão acessibilidade.



No arquivo *style.css*, logo abaixo do código da classe **.rotacao-botao**, criaremos a classe **.opcoes-acessibilidade { }** e adicionaremos a ela as propriedades **margin-top:10px; display: flex; flex-direction: column;**. O código deve ficar da seguinte forma:

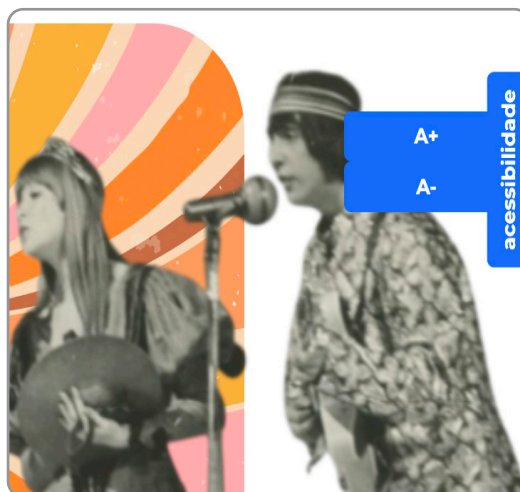
```
.rotacao-botao{  
    transform: rotate(-90deg);  
    transform-origin: right center;  
}  
  
.opcoes-acessibilidade{  
    margin-top:10px;  
    display: flex;  
    flex-direction: column;  
}
```

Feito isso, precisamos dizer, no arquivo HTML, que queremos usar essa nova classe.

Então, abriremos o arquivo *index.html* no VS Code e adicionaremos essa nova classe na **div id="opcoes-acessibilidade"**. Feito isso, o código ficará da seguinte forma:

```
<div id="acessibilidade" class="menu-acessibilidade">
  <button class="btn btn-primary fw-bold rotacao-botao" >acessibilidade</button>
  <div id="opcoes-acessibilidade" class="opcoes-acessibilidade">
    <button id="aumentar-fonte" class="btn btn-primary fw-bold">A+</button>
    <button id="diminuir-fonte" class="btn btn-primary fw-bold">A-</button>
  </div>
</div>
```

Ao salvar o projeto, teremos o seguinte resultado:





Agora as opções estão em formato de lista, mas ainda podem ser aperfeiçoadas.

Assim, adicionaremos uma margem entre os botões A+ e A-, porque eles estão muito colados. Para isso, selecionaremos especificamente o botão das opções, adicionando a classe **.opcoes-acessibilidade button** ao arquivo *style.css* e adicionaremos a propriedade **margin-bottom: 5px**. O código deve ficar da seguinte forma:

```
.opcoes-acessibilidade{  
    margin-top: 10px;  
    display: flex;  
    flex-direction: column;  
}
```

```
.opcoes-acessibilidade button{  
    margin-bottom: 5px;  
}
```

Feito isso, a visualização deve estar mais adequada agora:



Agora, trabalharemos no comportamento de esconder ou exibir os botões do menu de acessibilidade. Para isso, no arquivo *style.css*, adicionaremos a classe que manterá os botões escondidos.

Essa classe se chamará `.apresenta-lista { }`. Ela deve conter a propriedade `display: none;`. Observe:

```
.opcoes-acessibilidade button{
  margin-bottom: 5px;
}

.apresenta-lista{
  display: none;
}
```

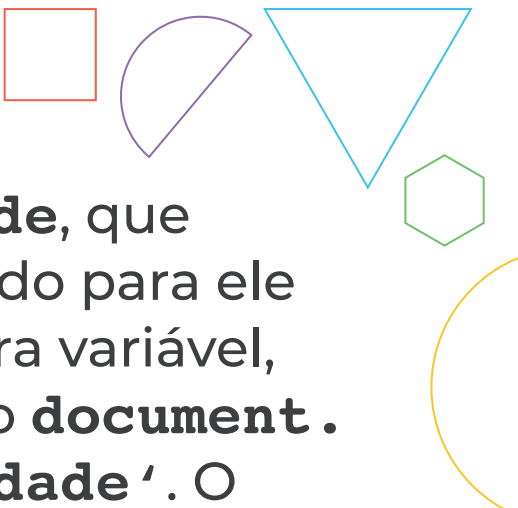
Feito isso, no arquivo `index.html`, precisamos chamar a nova classe de estilo na `div` com `id="opcoes-acessibilidade "`. Observe:

```
<div id="acessibilidade" class="menu-acessibilidade">
  <button id="botao-acessibilidade" class="btn btn-primary fw-bold
  rotacao-botao">acessibilidade</button>
  <div id="opcoes-acessibilidade" class="opcoes-acessibilidade
  apresenta-lista">
    <button id="aumentar-fonte" class="btn btn-primary fw-bold"
    >A+</button>
    <button id="diminuir-fonte" class="btn btn-primary fw-bold"
    >A-</button>
  </div>
</div>
```

Ao salvar o projeto e atualizar o navegador, veremos que os botões foram escondidos, como esperado:



Com os botões escondidos, precisamos configurar o clique no botão para acessar a lista. Para isso, alteraremos o arquivo *script.js*. Dentro da classe principal (**`document.addEventListener('DOMContentLoaded' , function() { }`**), abriremos um espaço no início para adicionar essa funcionalidade, focando o botão de acessibilidade.



Assim, criaremos a variável **const botaoDeAcessibilidade**, que receberá o método **document.getElementById()**, passando para ele o ID **'botao-acessibilidade'**. Além disso, criaremos outra variável, **const opcoesDeAcessibilidade**, que receberá o método **document.getElementById()**, passando o ID **'opcoes-acessibilidade'**. O código JavaScript deve ficar da seguinte forma:

```
document.addEventListener('DOMContentLoaded', function(){  
  ...  
  
  diminuirFonteBotao.addEventListener('click', function(){  
    tamanhoAtualFonte -= 0.1;  
    document.body.style.fontSize = `${tamanhoAtualFonte}rem`  
  
  })  
  const botaoDeAcessibilidade = document.getElementById('botao-acessibilidade');  
  const opcoesDeAcessibilidade = document.getElementById('opcoes-acessibilidade');  
})
```




Agora, precisamos saber quando o usuário clicou sobre o botão de acessibilidade. Então, chamaremos o **botaoDeAcessibilidade** e adicionaremos a ele o **addEventListener('click', function() {})**. Nessa função, vamos adicionar a seguinte linha de código **botaoDeAcessibilidade.classList.toggle('rotacao-botao')**. O **classList.toggle** fará toda a lógica de adicionar ou remover uma classe CSS dos botões.

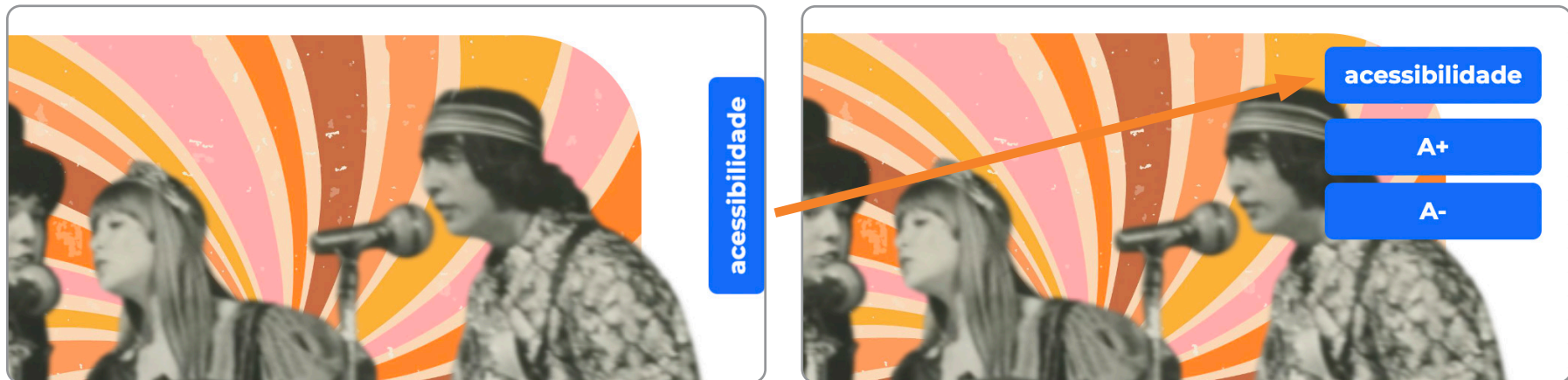
Se a classe já existe, ele retira, e se ela não existe, ele inclui!

Vamos fazer a mesma coisa para `opcoesDeAcessibilidade` e aplicar nelas `.classList.toggle('apresenta-lista')`.

```
document.addEventListener('DOMContentLoaded', function(){
  ...
  diminuirFonteBotao.addEventListener('click', function(){
    tamanhoAtualFonte -= 0.1;
    document.body.style.fontSize = `${tamanhoAtualFonte}rem`
  })
  const botaoDeAcessibilidade = document.getElementById('botao-
  acessibilidade');
  const opcoesDeAcessibilidade = document.getElementById('opcoes-
  acessibilidade');
  botaoDeAcessibilidade.addEventListener('click', function (){
    botaoDeAcessibilidade.classList.toggle('rotacao-botao');
    opcoesDeAcessibilidade.classList.toggle('apresenta-lista');
  })
})
```

Vamos salvar o projeto e verificar se funcionou no navegador.

Agora, quando clicamos no botão de acessibilidade, que está na vertical, ele fica na horizontal e exibe as opções de aumentar e diminuir o tamanho da fonte em uma lista.



► Desafio

Nesta aula, aprendemos a configurar os botões controlando sua exibição e posição na tela a partir do evento de clique e usando classes do CSS para isso. Que tal ir além e testar como poderíamos deixar o botão de acessibilidade em outras posições na tela? Ou mesmo fazer com que as opções de acessibilidade sejam exibidas ao passarmos o mouse sobre o botão acessibilidade? Dica: a posição do botão foi configurada no CSS, enquanto o evento de clique (ou de passar o mouse) deve ser configurado no JavaScript.



CLIQUE AQUI PARA AVALIAR ESTE MATERIAL