View Encapsulation: Shadow DOM

No Angular, o **Shadow DOM** é um modo de encapsulamento de visualização (view encapsulation) que utiliza a especificação do Shadow DOM do navegador. Ele isola o estilo e a estrutura do componente de forma que:

- O CSS do componente não vaza para outros componentes.
- O CSS externo não afeta o estilo do componente encapsulado.

Isso é útil para criar componentes verdadeiramente independentes e reutilizáveis.

Configuração do Encapsulamento `ShadowDom`

Para habilitar o encapsulamento no modo `ShadowDom`, você deve definir a propriedade `encapsulation` no decorador `@Component` como `ViewEncapsulation.ShadowDom`:

```
import { Component, ViewEncapsulation } from '@angular/core';

@Component({
  selector: 'app-card',
  templateUrl: './card.component.html',
  styleUrls: ['./card.component.scss'],
  encapsulation: ViewEncapsulation.ShadowDom
})
export class CardComponent {}
```

Como o Shadow DOM Funciona no Angular

1. **Isolamento de Estilos**
   O Angular cria um **shadow root** para o componente. O CSS dentro deste componente não interfere em outros componentes e também não é afetado pelo CSS global.

Estrutura do DOM

Quando você inspeciona um componente no modo `ShadowDom`, verá algo como:

```
<app-card>
 #shadow-root (open)
```

```
  <h1>Título do Card</h1>
  <p>Conteúdo do card</p>
</app-card>
```

2. **Compatibilidade**
   Funciona apenas em navegadores modernos que suportam Shadow DOM nativamente (como Chrome, Edge e Firefox). Nos navegadores mais antigos, você pode precisar de **polyfills**.

Exemplo Completo

Estrutura do Componente

`card.component.ts`

```
import { Component, ViewEncapsulation } from '@angular/core';

@Component({
  selector: 'app-card',
  templateUrl: './card.component.html',
  styleUrls: ['./card.component.scss'],
  encapsulation: ViewEncapsulation.ShadowDom
})
export class CardComponent {
  title = 'Exemplo Shadow DOM';
}
```

`card.component.html`

```
<div class="card">
  <h1>{{ title }}</h1>
  <p>Este é um exemplo usando Shadow DOM no Angular.</p>
</div>
```

`card.component.scss`

```
.card {
  background-color: lightblue;
  border: 2px solid darkblue;
  padding: 20px;
```

```
  border-radius: 8px;
}

h1 {
  color: darkblue;
}
```

Resultado

Se houver um componente global no projeto com estilos como:

```
h1 {
  color: red;
}
```

- **Sem Shadow DOM**: Os títulos `<h1>` do componente serão afetados pelo estilo global e ficarão vermelhos.
- **Com Shadow DOM**: O `<h1>` dentro de `app-card` permanecerá azul escuro (`darkblue`), conforme definido no CSS do componente, isolado do estilo global.

Recursos Avançados no Shadow DOM

1. Estilos Globais Não Penetram

Os estilos definidos no arquivo CSS global ou em outros componentes não afetam o componente encapsulado no modo `ShadowDom`.

2. Estilizando Elementos do Host

Use o seletor `:host` para aplicar estilos diretamente ao elemento `host` (raiz) do componente:

Exemplo

```
:host {
  display: block;
  width: 300px;
  border: 2px solid black;
}
```

Isso adicionará uma borda ao elemento `<app-card>`.

---

3. Estilizando o Elemento `Host` com Condições

Você pode usar o seletor `:host` com pseudo-classes:

Exemplo

```
:host(:hover) {
  border-color: blue;
}
```

Aqui, a borda do componente muda para azul quando o mouse está sobre o componente.

---

4. Usando `:host-context`

Se você quiser que o componente seja estilizado com base em um estilo do ambiente em que ele está inserido, use `:host-context`.

Exemplo

```
:host-context(.dark-theme) {
  background-color: black;
  color: white;
}
```

Isso aplica o estilo apenas se o componente estiver dentro de um contêiner com a classe `dark-theme`.

Limitações do Shadow DOM

1. **Complexidade no Debugging**
   Inspecionar elementos encapsulados pode ser mais complicado, pois os estilos ficam isolados.

2. **Impossibilidade de Estilos Globais**
   Não é possível aplicar estilos globais ao Shadow DOM sem hacks ou usar

um `::part`/`::theme`.

3. **Compatibilidade Limitada**
   Navegadores antigos podem não oferecer suporte ao Shadow DOM nativamente.

Quando Usar o Encapsulamento `ShadowDom`

1. **Componentes Reutilizáveis**
   Para bibliotecas de componentes (ex.: botões, sliders, etc.), onde o isolamento total dos estilos é desejado.

2. **Prevenção de Conflitos**
   Se o projeto tiver estilos globais que podem interferir nos componentes.

3. **Estilos Escaláveis**
   Projetos grandes, onde gerenciar a poluição de estilos globais seria difícil.

Resumo

O encapsulamento `ShadowDom` no Angular é uma ferramenta poderosa para criar componentes verdadeiramente independentes e isolados, ideal para bibliotecas e projetos escaláveis. Ele garante que os estilos e comportamentos sejam completamente encapsulados, embora exija atenção especial à compatibilidade com navegadores antigos e ao design de estilos globais.