

Раздел

Веб-компоненты

Навигация по уроку

Встроенный теневой DOM

Теневое дерево

Инкапсуляция

Ссылки


Итого

Комментарии

Поделиться



Редактировать на GitHub

 → Веб-компоненты 25-го сентября 2019

Shadow DOM

Теневой DOM («Shadow DOM») используется для инкапсуляции. Благодаря ему в компоненте есть собственное «теневое» DOM-дерево, к которому нельзя просто так обратиться из главного документа, у него могут быть изолированные CSS-правила и т.д.

Встроенный теневой DOM

Задумывались ли вы о том, как устроены и стилизованы сложные браузерные элементы управления?

Например, `<input type="range">`:



Браузер рисует их своими силами и по своему усмотрению. Их DOM-структура обычно нам не видна, но в инструментах разработчика можно её посмотреть. К примеру, в Chrome для этого нужно активировать пункт «Show user agent shadow DOM».

После этого `<input type="range">` выглядит так:

```
<input type="range"> == $0
#shadow-root (user-agent)
  <div>
    <div pseudo="-webkit-slider-runable-track" id="track">
      <div id="thumb"></div>
    </div>
  </div>
</input>
```

То, что находится под `#shadow-root` – и называется «shadow DOM» (теневой DOM).

Мы не можем получить доступ к теневому DOM встроенных элементов с помощью обычных JavaScript-вызовов или с помощью селекторов. Это не просто обычные потомки, это мощное средство инкапсуляции.

В примере выше можно увидеть полезный атрибут `pseudo`. Он нестандартный и существует по историческим причинам. С его помощью можно стилизовать подэлементы через CSS, например, так:

```
1 <style>
2 /* делаем цвет шкалы ползунка красным */
3 input::-webkit-slider-runable-track {
4   background: red;
5 }
6 </style>
7
8 <input type="range">
```



Ещё раз заметим, что `pseudo` – нестандартный атрибут. Если говорить хронологически, то сначала браузеры начали экспериментировать с инкапсуляцией внутренних DOM-структур для элементов, а уже потом, через некоторое время, появился стандарт Shadow DOM, который позволяет делать то же самое нам, разработчикам.

Далее мы воспользуемся современным стандартом Shadow DOM, описанным в спецификации [DOM spec](#) и других спецификациях.

Теневое дерево

Каждый DOM-элемент может иметь 2 типа поддеревьев DOM:

Раздел

Веб-компоненты

Навигация по уроку

Встроенный теневой DOM

Теневое дерево

Инкапсуляция

Ссылки

Итого

Комментарии

Поделиться



Редактировать на GitHub



1. Light tree – обычное, «светлое», DOM-поддерево, состоящее из HTML-потомков. Все поддеревья, о которых мы говорили в предыдущих главах, были «light».
2. Shadow tree – скрытое, «теневое», DOM-поддерево, не отражённое в HTML, скрытое от посторонних глаз.

Если у элемента имеются оба поддерева, браузер отрисовывает только теневое дерево. Также мы всё же можем задать «композицию» теневого и обычного деревьев. Позже в главе [Слоты теневого DOM, композиция](#) мы рассмотрим детали.

Теневое дерево можно использовать в пользовательских элементах (Custom Elements), чтобы спрятать внутренности компонента и применить к ним локальные стили.

Например, этот `<show-hello>` элемент прячет свой внутренний DOM в теневом дереве:

```
1 <script>
2   customElements.define('show-hello', class extends HTML
3     connectedCallback() {
4       const shadow = this.attachShadow({mode: 'open'});
5       shadow.innerHTML = `<p>
6         Hello, ${this.getAttribute('name')}
7       </p>`;
8     }
9   });
10 </script>
11
12 <show-hello name="John"></show-hello>
```

Hello, John



А вот как получившийся DOM выглядит в инструментах разработчика в Chrome, весь контент внутри «#shadow-root»:



```
▼ <show-hello name="John"> == $0
  ▼ #shadow-root (open)
    <p>Hello, John!</p>
  </show-hello>
```

Итак, вызов `elem.attachShadow({mode: ...})` создаёт теневое дерево.

Есть два ограничения:

1. Для каждого элемента мы можем создать только один shadow root.
2. В качестве `elem` может быть использован пользовательский элемент (Custom Element), либо один из следующих элементов: «article», «aside», «blockquote», «body», «div», «footer», «h1...h6», «header», «main», «nav», «p», «section» или «span». Остальные, например, ``, не могут содержать теневое дерево.

Свойство `mode` задаёт уровень инкапсуляции. У него может быть только два значения:

- "open" – корень теневого дерева («shadow root») доступен как `elem.shadowRoot`.

Любой код может получить теневое дерево `elem`.

- "closed" – `elem.shadowRoot` всегда возвращает `null`.

До теневого DOM в таком случае мы сможем добраться только по ссылке, которую возвращает `attachShadow` (и, скорее всего, она будет спрятана внутри класса). Встроенные браузерные теневые деревья, такие как у `<input type="range">`, закрыты. До них не добраться.

С возвращаемым методом `attachShadow` объектом [корнем теневого дерева](#), можно работать как с обычным DOM-элементом: менять его `innerHTML` или использовать методы DOM, такие как `append`, чтобы заполнить его.

Раздел

Веб-компоненты

Навигация по уроку

Встроенный теневого DOM

Теневое дерево

Инкапсуляция

Ссылки

Итого

Комментарии

Поделиться



Редактировать на GitHub

Элемент с корнем теневого дерева называется – «хозяин» (host) теневого дерева, и он доступен в качестве свойства `host` у `shadow root`:

```
1 // при условии, что {mode: "open"}, иначе elem.shadowRoot
2 alert(elem.shadowRoot.host === elem); // true
```



Инкапсуляция

Теневой DOM отделён от главного документа:

1. Элементы теневого DOM не видны из обычного DOM через `querySelector`. В частности, элементы теневого DOM могут иметь такие же идентификаторы, как у элементов в обычном DOM (light DOM). Они должны быть уникальными только внутри теневого дерева.
2. У теневого DOM свои стили. Стили из внешнего DOM не применяются.

Например:

```
1 <style>
2   /* стили документа не применяются в теневом дереве вну
3   p { color: red; }
4 </style>
5
6 <div id="elem"></div>
7
8 <script>
9   elem.attachShadow({mode: 'open'});
10   // у теневого дерева свои стили (2)
11   elem.shadowRoot.innerHTML = `
12     <style> p { font-weight: bold; } </style>
13     <p>Hello, John!</p>
14   `;
15
16   // <p> виден только запросам внутри теневого дерева (
17   alert(document.querySelectorAll('p').length); // 0
18   alert(elem.shadowRoot.querySelectorAll('p').length);
19 </script>
```



1. Стили главного документа не влияют на теневое дерево.
2. ...Но свои внутренние стили работают.
3. Чтобы добраться до элементов в теневом дереве, нам нужно искать их изнутри самого дерева.

Ссылки

- DOM: <https://dom.spec.whatwg.org/#shadow-trees>
- Совместимость: <https://caniuse.com/#feat=shadowdomv1>
- Теневого DOM упоминается во многих других спецификациях, например [DOM Parsing](#) указывает, что у `shadow root` есть `innerHTML`.

Итого

Теневой DOM – это способ создать свой, изолированный, DOM для компонента.

1. `shadowRoot = elem.attachShadow({mode: open|closed})` – создаёт теневого DOM для `elem`. Если `mode="open"`, он доступен через свойство `elem.shadowRoot`.
2. Мы можем создать подэлементы внутри `shadowRoot` с помощью `innerHTML` или других методов DOM.

Элементы теневого DOM:

- Обладают собственной областью видимости идентификаторов
- Невидимы JavaScript селекторам из главного документа, таким как `querySelector`,

Раздел

[Веб-компоненты](#)

Навигация по уроку

Встроенный теневого DOM

Теневое дерево

Инкапсуляция

Ссылки

Итого

Комментарии

Поделиться



Редактировать на GitHub

- Стилизируются своими стилями из теневого дерева, не из главного документа.



Теневой DOM, если имеется, отрисовывается браузером вместо обычных потомков (light DOM). В главе [Слоты теневого DOM, композиция](#) мы разберём, делать их композицию.

Проводим [курсы по JavaScript и фреймворкам](#).



Комментарии

перед тем как писать...

© 2007—2020 Илья Кантор | [о проекте](#) | [связаться с нами](#) | [пользовательское соглашение](#) | [политика конфи](#)

