Front-end

JavaScript

JS. data attributes

- data attributes
- dataset

JS. data attributes

Все атрибуты у HTML элементов, начинающиеся с префикса data-*, являются пользовательскими. Data атрибуты можно использовать для дополнительной стилизации, но чаще всего они применяются для создания интерактивной логики в JavaScript. Особенно их любят применять разные библиотеки, когда пользователю предлагается настроить скрипт через data атрибуты.

JS. data attributes

Data атрибут - это очень гибкий инструмент и сейчас мы рассмотрим, как можно его использовать в HTML и CSS.

Как добавить data атрибут к HTML тегу

Вначале обязательно ставим префикс data, затем через дефис указываем какое-то наше слово подходящее по смыслу и само значение. Например мы хотим отсортировать только категорию с домашними питомцами. Все слова, за исключением самого префикса data-*, мы можем придумывать свои собственные. Так мы можем управлять отдельными группами элементов, помеченные data атрибутами. Это удобно для создания интерактива на языке JavaScript.

JS. data attributes

<button data-categories="pets">Pets</button>

Длина названия data атрибута может быть любой, слова должны разделяться через дефис.

JS. data attributes

Мы можем стилизовать любой элемент по его data атрибуту. В CSS коде селектор data атрибута заключается в квадратные скобки. Обращаться можно только по названию атрибута, по тегу + название или по классу (id) + название.

```
<div class="large_btn" data-size="large">button</div>

[data-size="large"] {  // селектор по названию атрибута
   font-size: 30px;
   padding: 20px;
}
div [data-size="large"] {  // селектор по тегу и названию
   font-size: 30px;
   padding: 20px;
}
large_btn.[data-size="large"] {  // селектор по классу и названию
   font-size: 30px;
   padding: 20px;
}
```

JS. data attributes

Принцип создания подсказок с data атрибутом на CSS

Прописываем текст подсказки в data атрибуте тега. Затем с помощью псевдоэлементов ::before или ::after передать в функцию attr значение атрибута data-tooltip.

```
<span data-tooltip="текст подсказки"></span>
```

```
[data-tooltip]::after {
   content: attr(data-tooltip);
}
```

JS. data attributes

Использование data атрибутов в JavaScript

В JS существует, как минимум два способа получения data атрибута:

Первый способ, через getAttribute и setAttribute

Указываем выбранный на странице элемент (тег, id, класс), сам метод getAttribute и название атрибута, который надо получить.

```
element.getAttribute("data-filter")
```

Метод setAttribute добавляет значение, указанное вторым параметром ("pillows") в data атрибут в первом параметре ("data-filter").

```
element.setAttribute("data-filter", "pillows")
```

JS. data attributes

Второй способ, через объект dataset в структуре DOM

Доступ к коллекции dataset мы получаем по ключу (слово после префикса data-*).

```
<div data-person></div>

// Получение data атрибута
div.dataset.person

// Добавление значения для data атрибута
div.dataset.person = "Donald"
```

JS. data attributes

Свойство dataset позволяет считывать или устанавливать любые дата-атрибуты на HTML-элементе.

Дата-атрибут — это пользовательский атрибут на элементе, название которого начинается с data-, например data-testid. Дата атрибуты используются, чтобы хранить значения на элементах в HTML.

Как пишется

Обращение к свойству dataset вернёт объект со всеми дата-атрибутами, которые есть на элементе. Названиями полей в объекте будут имена дата-атрибутов после префикса data-. Например, если атрибут называется data-columns, то поле в объекте для этого атрибута будет называться columns.

JS. data attributes

```
<h1>Известные ситхи</h1>
<ul>
 <li data-id="1541" data-episode="1">Дарт Мол</li>
 <li data-id="9434" data-episode="4">Дарт Вейдер</li>
 <li data-id="5549" data-episode="4">Дарт Сидиус</li>
</ul>
const items = document.querySelectorAll('li')

const firstItem = items[0]

console.log(firstItem.dataset)
// { id: '1541', episode: '1' }
```

JS. data attributes

Если дата-атрибутов на элементе нет, то вернётся пустой объект:

```
const heading = document.querySelector('h1')

console.log(heading.dataset)
// {}
```

JS. data attributes

Чтобы добавить дата-атрибут к элементу, нужно добавить новое поле в объект dataset. Название поля так же должно быть без префикса data-, браузер автоматически подставит его. В значениях атрибутов в HTML могут быть только строки, потому любое значение будет автоматически приведено к строке.

Возьмём тот же HTML из примера выше и добавим дата-атрибуты ко второму элементу:

```
const items = document.querySelectorAll('li')

const secondItem = items[1]

secondItem.dataset.side = 'evil'
secondItem.dataset.age = 46
secondItem.dataset.lightsaber = { color: 'red' }
```

JS. data attributes

В результате получим такой элемент:

```
<li
 data-id="9434"
 data-episode="4"
 data-side="evil"
 data-age="46"
 data-lightsaber="[object Object]">
 Дарт Вейдер
</li>
```

JS. data attributes

Использование camelCase и kebab-case

В dataset необходимо присваивать поля, название которых записывается в одно слово. Потому для составных имён используется только camelCase нотация. При попытке присвоить название в kebab-case будет выброшена ошибка.

```
const body = document.querySelector('body')

body.dataset['dark-theme'] = true

// Uncaught DOMException: Failed to set
// a named property on 'DOMStringMap':
// 'dark-theme' is not a valid property name.
```

JS. data attributes

Дата-атрибуты, записанные в dataset с помощью camelCase, в HTML будут иметь названия в kebab-case. Браузер преобразует camelCase в kebab-case:

```
<ul>
  <li>Иван Иванов</li>
</ul>

const item = document.querySelector('li')

item.dataset.yearsOfExperience = 2
item.dataset.candidateRole = 'junior'
```

После выполнения кода выше получится следующий HTML:

```
<ul>
  <li data-candidate-role="junior" data-years-of-experience="2">Иван Иванов</li>
</ul>
```

JS. data attributes

Удаление дата-атрибута

Удалить дата-атрибут можно только с помощью оператора delete. Если попытаться присвоить к полю значение undefined или null, то браузер просто присвоит атрибуту строку 'undefined' или 'null'.

Возьмём следующий HTML:

Любое содержимое

При установке undefined в значение дата-атрибута, он не удалится с элемента.

const element = document.querySelector('div')

element.dataset.testid = undefined

В результате получится следующий HTML:

Любое содержимое

JS. data attributes

Если использовать оператор delete, то получим элемент без дата-атрибута.

```
delete element.dataset.testid
<div>Любое содержимое</div>
```

Свойство dataset защищено от перезаписи. Это значит что попытка присвоить в dataset новое значение будет проигнорирована.

```
const element = document.querySelector('div')

// Ничего не произойдёт, дата-атрибуты на элементах тоже не изменятся
element.dataset = {}
element.dataset = 'string'
```

JS. data attributes

Data атрибуты позволяют хранить разную информацию об элементе, которая может помочь для работы скриптов, а также для CSS стилизации элементов. HTML код с созданными атрибутами с data-* префиксом будет, абсолютно валидным. Создавать свои собственные data атрибуты для хранения значений стало возможным лишь в HTML5, до этого такой возможности очень не хватало веб-разработчикам. Вот список самых востребованные задач, которые удобно решать с помощью data атрибутов:

Создание всплывающих подсказок на чистом CSS

Получать и изменять значения атрибутов

JS. data attributes

Дата-атрибуты широко используются в **автоматизированном тестировании**. Для этого на необходимых элементах расставляют дата-атрибуты и тест обращается к ним. В документациях к различным библиотекам для тестирования часто можно встретить атрибут **data-testid**