WEB. Основы Front-End разработки.

Урок 22. Теория. Дополнительные инструменты работы с DOM.

Дополнительные конструкции.

В JavaScript часто приходится работать с DOM (Document Object Model) для взаимодействия с элементами веб-страницы. Рассмотрим некоторые, упрощающие работу, конструкции и другие полезные методы.

Проверка привязки класса: event.target.classList.contains().

Метод event.target.classList.contains(className) проверяет, содержит ли элемент, который вызвал событие, определенный CSS-класс.

Пример:

```
document.addEventListener('click', function(event) {
    if (event.target.classList.contains('my-class')) {
        console.log('Был выбран элемент с классом "my-class"');
    }
});
```

В этом примере, когда пользователь кликает на страницу, проверяется, содержит ли кликнутый элемент класс my-class. Если да, то в консоль выводится сообщение.

Поиск ближайшего элемента: event.target.closest().

Метод event.target.closest(selector) возвращает ближайший родительский элемент (или сам элемент), который соответствует заданному CSS-селектору. Если такой элемент не найден, возвращает null.

Пример:

```
document.addEventListener('click', function(event) {
    let closestElement = event.target.closest('.container');
    if (closestElement) {
        console.log('Элемент с классом "container" найден');
    }
});
```

Здесь, если кликнутый элемент или любой из его родителей имеет класс container, то в консоль выводится сообщение.

Поиск по селектору: querySelector().

querySelector(selector) возвращает первый элемент документа, который соответствует заданному CSS-селектору. Если таких элементов нет, возвращает null.

Пример:

```
let element = document.querySelector('.my-class');
if (element) {
    console.log('Элемент с классом "my-class" найден');
}
```

В этом примере ищется первый элемент с классом my-class и выводится сообщение, если такой элемент найден.

Дополнительные конструкции для работы с DOM.

Метод querySelectorAll().

Возвращает все элементы в документе, которые соответствуют заданному селектору. Результат - NodeList.

Пример:

```
let elements = document.querySelectorAll('.my-class');
elements.forEach(el => {
    console.log(el);
});
```

Свойство innerHTML.

Свойство innerHTML позволяет получать или устанавливать HTML-содержимое элемента. Оно возвращает строку, представляющую HTML, который содержится в элементе, или позволяет установить HTML, который будет отрендерен в элементе.

Получение HTML-содержимого.
Чтобы получить HTML-содержимое элемента, просто используйте innerHTML без параметров:

```
let element = document.querySelector('#my-element');
console.log(element.innerHTML);
```

Этот код выведет HTML-содержимое элемента с id="my-element" в консоль.

Установка HTML-содержимого.
Чтобы установить HTML-содержимое элемента, присвойте innerHTML новую строку:

```
let element = document.querySelector('#my-element');
element.innerHTML = '<p>New content</p>';
```

Этот код заменит текущее содержимое элемента с id="my-element" на New content .

Комбинирование с существующим содержимым.

Если вы хотите добавить новое содержимое, сохраняя текущее, вы можете комбинировать innerHTML:

```
let element = document.querySelector('#my-element');
element.innerHTML += '<p>Additional content</p>';
```

Это добавит Additional content в конец текущего содержимого элемента.