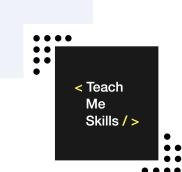
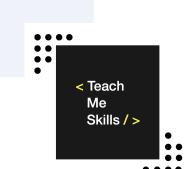
Events

- Браузерные события
- Обработчики событий
- Принципы всплытия и погружения
- Делегирование событий



Урок

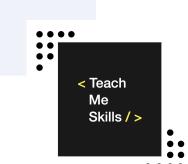




Событие – это сигнал от браузера о том, что что-то произошло.

Часто используемые события в браузере:

- События на элементах управления
- События мыши
- Клавиатурные события
- События документа
- CSS events



Событию можно назначить обработчик, то есть функцию, которая сработает, как только событие произошло.

Именно благодаря обработчикам JavaScript-код может реагировать на действия пользователя.

Есть три способа назначения обработчиков событий

Через атрибут HTML

DOM свойство onclick у элемента

Метод addEventListener



Через атрибут HTML

HTML-атрибуты используются редко потому, что JavaScript в HTML-теге выглядит немного странно. К тому же много кода там не напишешь.

```
<button onclick="alert('Hi!')"></button>
```



DOM свойство onclick у элемента

DOM-свойства вполне можно использовать, но мы не можем назначить больше одного обработчика на один тип события. Во многих случаях с этим ограничением можно мириться.

```
const btn = document.getElementById('button')
btn.onclick = function() {
   console.log('Hi!');
}
```



Mетод addEventListener

Обладает более широким функционалом. Позволяет добавлять несколько обработчиков на одно событие одного элемента. Удалять слушатели а также отслеживать дополнительные события.

```
const btn = document.getElementById('button')
btn.addEventListener('click', function(){
    console.log('Hi');
})
```



Mетод addEventListener

Событие: клик, скролл, нажатия клавиатуры и тд.

target.addEventListener(event, callback, options)

Функция которую нужно вызвать, __ при совершении события.

Дополнительный не обязательный объект со свойствами.



Объект события

Когда происходит событие, браузер создаёт объект события, записывает в него детали и передаёт его в качестве аргумента функции-обработчику.

```
const btn = document.getElementById('button')
btn.addEventListener('click', function(event){
    console.log(event);
})
```



Принципы всплытия и погружения

Event bubbling Event capturing <body> <div> Clicked! **Event target**



Делегирование событий

<l

Если у нас есть много элементов, события на которых нужно обрабатывать похожим образом, то вместо того, чтобы назначать обработчик каждому, мы можем воспользоваться делегированием событий.

Шаг 1: Определить родителя элементов для отслеживания событий

Шаг 2: Прикрепить на него обработчик событий

Шаг 3: Использовать event.target для выбора целевого элемента

