События DOM

События мыши:

- **click** – происходит, когда кликнули на элемент левой кнопкой мыши (на устройствах с сенсорными экранами оно происходит при касании).
- **contextmenu** – происходит, когда кликнули на элемент правой кнопкой мыши.
- **mouseover** / **mouseout** – когда мышь наводится на / покидает элемент.
- **mousedown** / mouseup – когда нажали / отжали кнопку мыши на элементе.
- **mousemove** – при движении мыши.

События на элементах управления:

- **submit** – пользователь отправил форму <form>.
- **focus** – пользователь фокусируется на элементе, например нажимает на <input>.

Клавиатурные события:

keydown и **keyup** – когда пользователь нажимает / отпускает клавишу.

События документа:

DOMContentLoaded – когда HTML загружен и обработан, DOM документа полностью построен и доступен.

Использование атрибута HTML:

```
<input type="button” value="Нажми меня" onclick="alert('Клик!’)”>
```

или

```
<script>
  function doSomething () {
                    alert('Клик!’);
      }
  </script>

<input type="button" value="Нажми меня" onclick="doSomething()" >
```

Использование свойства DOM-объекта (не ркомендуется)

```
<input id="b1" type="button" value="Нажми меня!">
<script>
let elem = document.getElementById("b1");
elem.onclick = function() {
  alert("Спасибо");
};
</script>
```

или

```
<input id="b1" type="button" value="Нажми меня!">
<script>
function sayThanks() {
  alert('Спасибо!');
}
let elem = document.getElementById("b1");
elem.onclick = sayThanks;
</script>
```

У элемента DOM может быть только одно свойство с именем onclick

```
<input type="button" id="elem" onclick="alert('Было')" >
<script>
  elem.onclick = function() { // перезапишет существующий обработчик
    alert('Станет'); // выведется только это
  };
</script>
```

Убрать обработчик можно назначением `elem.onclick = null`.

Внутри обработчика события this ссылается на текущий элемент:

```
<button onclick="alert(this.type)">Нажми меня</button>
```

addEventListener

element.addEventListener(type, handler[, options]);

- **type**- Тип (имя) события, например "click".
- **handler** - Ссылка на функцию-обработчик.
- **options** - Дополнительный объект со свойствами:
  - once: если true, тогда обработчик будет автоматически удалён после выполнения.
  - capture: фаза, на которой должен сработать обработчик,
  - passive: если true, то указывает, что обработчик никогда не вызовет preventDefault

Для удаления обработчика следует использовать removeEventListener

Объект события

- **event.type** - Тип события, например "click".
- **event.currentTarget** - Элемент, на котором сработал обработчик. Значение – обычно такое же, как и у this
- **event.target** - Самый глубокий элемент, который вызывает событие
- **event.clientX** / **event.clientY -** Координаты курсора в момент клика относительно окна, для событий мыши.

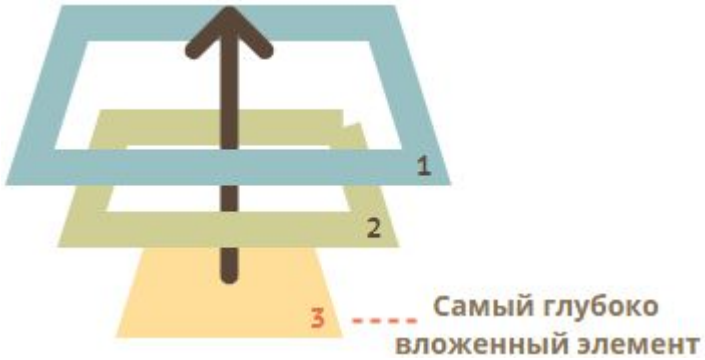
```
<input type="button" value="Нажми меня" id="elem">
<script>
  elem.onclick = function(event) {
    // вывести тип события, элемент и координаты клика
    alert(event.type + " на " + event.currentTarget);
    alert("Координаты: " + event.clientX + ":" + event.clientY);
  };
</script>
```

Всплытие

Когда на элементе происходит событие, обработчики сначала срабатывают на нём, потом на его родителе, затем выше и так далее, вверх по цепочке предков.

```
<form onclick="alert('form')">FORM
  <div onclick="alert('div')">DIV
    <p onclick="alert('p')">P</p>
  </div>
</form>
```

https://codepen.io/SergMt/pen/QWWBOgg



https://plnkr.co/edit/Yaw0r5RodOsHLhGyeD62?p=preview

Прекращение всплытия

метод event.stopPropagation().

```
<body onclick="alert(`сюда всплытие не дойдёт`)">
  <button onclick="event.stopPropagation()">Кликни меня</button>
</body>
```

Делегирование событий

Примеры:
https://codepen.io/SergMt/pen/xxxJpMR
https://codepen.io/SergMt/pen/LYYBewj

Метод elem.closest(selector) возвращает ближайшего предка, соответствующего селектору. В данном случае нам нужен <td>, находящийся выше по дереву от исходного элемента.

Действия браузера по умолчанию

Отмена действия браузера:

- Основной способ – это воспользоваться объектом event. Для отмены действия браузера существует стандартный метод event.preventDefault().
- Если же обработчик назначен через on<событие>, то можно вернуть false из обработчика.

События, вытекающие из других

Некоторые события естественным образом вытекают друг из друга. Если мы отменим первое событие, то последующие не возникнут.

Например, событие mousedown для поля <input> приводит к фокусировке на нём и запускает событие focus. Если мы отменим событие mousedown, то фокусирования не произойдёт.

События мыши

- **mousedown/mouseup** - Кнопка мыши нажата/отпущена над элементом.
- **mouseover/mouseout** - Курсор мыши появляется над элементом и уходит с него.
- **mousemove** - Каждое движение мыши над элементом генерирует это событие.
- **contextmenu** - Вызывается при попытке открытия контекстного меню, как правило, нажатием правой кнопки мыши.

Комплексные события

- **click** - Вызывается при mousedown , а затем mouseup над одним и тем же элементом, если использовалась левая кнопка мыши.
- **dblclick** - Вызывается двойным кликом на элементе.

свойство button

event.button == 0 – левая кнопка
event.button == 1 – средняя кнопка
event.button == 2 – правая кнопка

еще есть event.buttons (https://codepen.io/Serg-Mt/pen/XWYqyEM)
event.which - устаревшее свойство

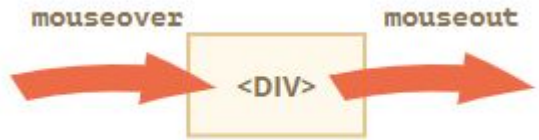
Свойства объекта события:

- **shiftKey**: Shift
- **altKey**: Alt (или Opt для Mac)
- **ctrlKey**: Ctrl
- **metaKey**: Cmd для Mac

Координаты:

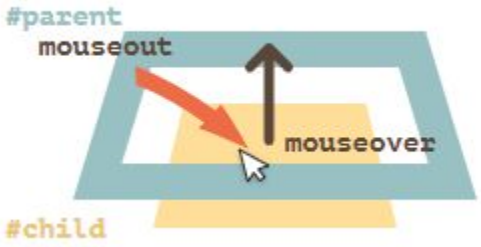
Все события мыши имеют координаты двух видов:

Относительно окна: clientX и clientY.
Относительно документа: pageX и pageY.



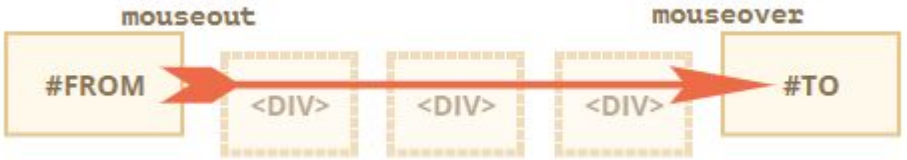
Для события **mouseover**:
event.target – это элемент, на который курсор перешёл.
event.relatedTarget – это элемент, с которого курсор ушёл (relatedTarget → target).

Для события **mouseout** наоборот:
event.target – это элемент, с которого курсор ушёл.
event.relatedTarget – это элемент, на который курсор перешёл (target → relatedTarget).



События mouseover/out возникают, даже когда происходит переход с родительского элемента на потомка.

Свойство relatedTarget может быть null.
Если был mouseover, то будет и mouseout



События **mouseenter/mouseleave**

- Переходы внутри элемента, на его потомки и с них, не считаются.
- События **mouseenter/mouseleave** не всплывают.

Свойства и методы формы

Событие **focus** вызывается в момент фокусировки,
blur – когда элемент теряет фокус.
Они не всплывают. Но можно использовать **focusin**/**focusout**

Событие **change** - срабатывает по окончании изменения элемента.
Событие **input** - срабатывает каждый раз при изменении значения.
События: **cut**, **copy**, **paste** - происходят при вырезании/копировании/вставке данных.