

Генерація подій

Генерация события

Event `event = new Event(typeArg, eventInit);`

- Тип события – может быть как своим, так и встроенным, к примеру "click"
- Флаги – объект вида { bubbles: true/false, cancelable: true/false }...

```
1 | var ev = new Event("look", {"bubbles":true, "cancelable":false});  
2 | document.dispatchEvent(ev);
```

Генерация события

dispatchEvent

- Отправляет событие в общую систему событий
- Событие подчиняется тем же правилам поведения (захват и всплытие) как и непосредственно инициированные события

```
1 cancelled = !target.dispatchEvent(event)
2 //event - объект события, который инициализируется
3 //target - используется для инициализации Event.target
4 //и установки события, которое обработчик вызывает
5
6 //return - false если хотя бы один из обработчиков
7 //этого события вызвал preventDefault
```

```
1 <button id="elem" onclick="alert('Клик');">Автоклик</button>
2 <script>
3   var event = new Event("click");
4   elem.dispatchEvent(event);
5 </script>
```

Генерация события

CustomEvent

- идентичен Event(), НО у второго аргумента-объекта есть дополнительное свойство detail, в котором можно указывать дополнительную информацию

```
1 <h1 id="elem">Element</h1>
2
3 <script>
4 var elem = document.getElementById('elem');
5   elem.addEventListener("hello", function(event) {
6     console.log( event.detail.name );
7   }, false);
8
9   var event = new CustomEvent("hello", {
10     detail: { name: "something" }
11   });
12
13   elem.dispatchEvent(event);
14 </script>
```

Таймеры

Таймеры в js

Таймеры - это не sleep(), они создают события, которые используют Event Loop

- `setTimeout(function, timeout)` - не ранее чем через timeout
- `setInterval(function, timeout)` - не чаще чем через timeout

Таймеры в js

```
function printMessage()
```

```
{ alert('Hello'); }
```

```
setTimeout(sayHi, 1000); // функция отработает с задержкой в  
1000ms
```

```
var foo = setInterval(() => console.log('Hello'), 1500); // функция  
отрабатывает циклически каждые 1500ms
```