이벤트 생성 및 트리거

이 글은 DOM 이벤트를 생성하고 디스패치하는 방법에 대해 설명합니다. 이런 이벤트는 흔히 **인공 이벤트(synthetic events)**라고 불리며, 브라우저 자체에서 실행되는 이벤트와 반대입니다.

커스텀 이벤트 생성하기

다음과 같이 `Event` 생성자를 사용해 Events 를 생성할 수 있습니다.

```
var event = new Event('build');

// 이벤트 리슨.
elem.addEventListener('build', function (e) { /* ... */ }, false);

// 이벤트 디스패치.
elem.dispatchEvent(event);
```

위 코드 예제는 EventTarget.dispatchEvent() 메소드를 사용합니다.

이 생성자는 대부분의 최신 브라우저(Internet Exploere 는 예외)에서 지원됩니다. 더 장황한 접근법(Internet Explorer 에서도 동작하는)은, 아래 옛날 방식 부분을 참고하세요.

커스텀 데이터 추가 – CustomEvent()

이벤트 객체에 더 많은 데이터를 추가하려면, CustomEvent 인터페이스가 존재하고 **detail** 프로퍼티를 통해 커스텀 데이터를 전달할 수 있습니다
예를 들면, 다음과 같이 이벤트가 생성될 수 있습니다.

```
var event = new CustomEvent('build', { detail: elem.dataset.time });
```

그럼 이벤트 리스너의 부가적인 데이터에 접근할 수 있게 됩니다.

```
function eventHandler(e) {
  console.log('The time is: ' + e.detail);
}
```

옛날 방식

생성 이벤트로의 오래된 접근법은 Java 로부터 영감을 받은 API들을 사용합니다. 다음은 그 예시를 보여줍니다.

```
// 이벤트 생성.
var event = document.createEvent('Event');

// 이벤트 이름을 'build' 라 정의.
event.initEvent('build', true, true);

// 이벤트 리슨.
elem.addEventListener('build', function (e) {
  // e.target 은 elem 과 일치
}, false);

// target 은 어떤 엘리먼트나 다른 이벤트 타켓이 될 수 있음.
elem.dispatchEvent(event);
```

이벤트 버블링

자식 엘리먼트로부터 이벤트를 발생시키고 그 조상이 이를 캐치하도록 하는것은 종종 바람직합니다. 선택적으로 데이터도 함께합니다.

```
<form>
  <textarea></textarea>
</form>
```

```
const form = document.querySelector('form');
const textarea = document.querySelector('textarea');

```

```
// 새로운 이벤트를 생성하고, 버블링을 허용하며, "details" 프로퍼티로 전달할
const eventAwesome = new CustomEvent('awesome', {
  bubbles: true,
  detail: { text: () => textarea.value }
});

// form 엘리먼트는 커스텀 "awesome" 이벤트를 리슨한 후 전달된 text() 메소
form.addEventListener('awesome', e => console.log(e.detail.text()));

// 사용자가 입력한대로, form 내의 textarea 는 이벤트를 디스패치/트리거하여
textarea.addEventListener('input', e => e.target.dispatchEvent(event
```

이벤트를 동적으로 생성하고 디스패칭하기

엘리먼트는 아직 생성되지 않은 이벤트를 리슨할 수 있습니다.

```
<form>
  <textarea></textarea>
</form>
```

```
const form = document.querySelector('form');
const textarea = document.querySelector('textarea');

form.addEventListener('awesome', e => console.log(e.detail.text()));

textarea.addEventListener('input', function() {
  // 이벤트 즉시 생성 및 디스패치/트리거
  // 노트: 선택적으로, 우리는 "함수 표현"("화살표 함수 표현" 대신)을 사용하
  this.dispatchEvent(new CustomEvent('awesome', { bubbles: true, deta
});
```

내장 이벤트 트리거

이 예제는 DOM 메소드를 사용해 체크박스에 클릭을 시뮬레이팅하는 것을 보여줍니다(클릭 이벤트를 프로그래밍적으로 발생시키는 것입니다). 동작하는 예제를 확인하세요.

```
function simulateClick() {
  var event = new MouseEvent('click', {
    view: window,
    bubbles: true,
    cancelable: true
  });
  var cb = document.getElementById('checkbox');
  var cancelled = !cb.dispatchEvent(event);
  if (cancelled) {
    // 핸들러가 preventDefault 를 호출했음.
    alert("cancelled");
  } else {
    // 어떤 핸들러도 preventDefault 를 호출하지 않음.
    alert("not cancelled");
  }
}
```

함께 보기

- CustomEvent()
- document.createEvent()
- Event.initEvent()
- EventTarget.dispatchEvent()
- EventTarget.addEventListener()

마지막 변경일: 2019년 3월 18일, by MDN contributors

뉴스레터는 아직 영어로만 제공됩니다.

you@example.com

지금 가입하기