Event

Ba Nguyễn

Introduction

💡 Sự kiện (event) là tín hiệu khi một điều gì đó xảy ra (trên DOM), tất cả các node trong DOM đều có thể kích hoạt các event (event không bị giới hạn trong DOM)

Để cung cấp một hành động phản ứng lại khi một sự kiện được kích hoạt, chúng ta sử dụng các trình xử lý sự kiện (event handlers)

💡 Event handler là một function được chạy khi một event được kích hoạt

Các cách để chỉ định một handler cho một event cụ thể:

- HTML Attribute
- DOM Property
- addEventListener()/removeEventListener()

HTML Attributes

Các handlers có thể gán cho một event cụ thể trên một element thông qua attributes **on*<event>***

Ví dụ:

```
<body onkeyup="greeting()">
    <p onclick="alert('Hello babe 🧡')">Click me!</p>
    <p onmouseover="let x = 1, y = 2; alert(x + y)">Hover me</p>
    <script>
      function greeting() {
        alert("Hello babe 😍");
      }
    </script>
</body>
```

DOM Properties

Tương tự attributes, có thể gán các handlers thông qua DOM properties **on*<event>***

Ví dụ:

```
p.onclick = function () { alert("Hello babe 🧡"); };
p.onmouseover = function () {
  let x = 1, y = 2;
  alert(x + y);
};
function greeting() { alert("Hello babe 😍"); }
document.body.onkeyup = function () {
  greeting();
};
```

Element - this

Giá trị của **this** trong handler chính là element kích hoạt event

```
<p onclick="alert(this.textContent)">Click me!</p> <!-- Click me! -->
```

!!!!!! Lưu ý về **this** khi sử dụng arrow function

```
p.onclick = () => alert(this.textContent); // undefined
```

💡 Khi gán một handler qua attribute lưu ý thêm dấu (), gán qua property thì không

💡 Không thể gán một handler thông qua phương thức `setAttribute()`

EventListener

💡 Chỉ có thể có một handler duy nhất cho một event khi gán thông qua attribute hoặc property

💡 Một số event đặc biệt không thể gán thông qua attribute hoặc property

JavaScript cung cấp 2 phương thức đặc biệt `addEventListener()/removeEventListener()` để quản lý các event tốt hơn, không gặp phải các hạn chế giống như attribute hoặc property.

Cú pháp:

```
el.addEventListener(event, handler [, options]);
el.removeEventListener(event, handler [, options]);
```

Options:

```
capture: false, passive: false, once: false
```

EventListener

Ví dụ:

```
let handler = function () {
  alert(this.textContent);
};

p.addEventListener("click", handler);

p.removeEventListener("click", handler);
```

💡 Lưu ý handler là giá trị hàm, không phải cuộc gọi hàm

💡 Khi thêm options trong `addEventListener()` thì `removeEventListener()` cũng cần truyền options tương ứng

EventObject

Handlers được gọi với một tham số đặc biệt - *event*, *event* chứa tất cả thông tin về sự kiện, như phần tử nào kích hoạt, phím được nhấn, ...

```
p.addEventListener("click", (e) => {
  alert(e.currentTarget); // p
  alert(e.currentTarget.textContent);
  alert(e.type); // click
});
```

💡 Đối với arrow function, `e.currentTarget` tương tự `this`

💡 Lưu ý `removeEventListener` yêu cầu cùng một phương thức (*tham chiếu*)