Lecture-14

Events

AJAX-API

Coding Blocks - Kartik Mathur

Class Agenda

EVENT LISTENERS!

JavaScript allows handling events like clicks, mouse movements, key presses, etc.
Two common ways to attach events:

1. `onClick` (Inline or DOM Property Event)
2. `addEventListener` (Event Listener Method)

What is `onClick`?

- Directly assigns an event handler to an element.
- Can only have **one** function assigned at a time (overwrites previous ones).
- Simple but **not flexible** for multiple events.

```
const btn = document.querySelector("button");
btn.onclick = function () {
   console.log("Button clicked!");
};
```

What is `addEventListener`?

- Allows adding multiple event listeners to the same element.
- More **flexible** and supports different event types.
- Can be **removed** using `removeEventListener`.

```
const btn = document.querySelector("button");
btn.addEventListener("click", () => {
   console.log("Button clicked!");
});
```

Overwriting Issue in onClick

- onClick **overwrites** the previous event.

```
btn.onclick = () => {
   console.log("First event");
};
btn.onclick = () => {
   console.log("Second event"); // This will replace the first one!
};
```

`addEventListener` **allows multiple handlers**.

```
btn.addEventListener("click", () => {
    console.log("First event");
});
btn.addEventListener("click", () => {
    console.log("Second event"); // Both will execute!
});
```

Event Listeners: To do a task on some event.

```
btn.addEventListener('click', () => {
   console.log("Button clicked!");
});
```

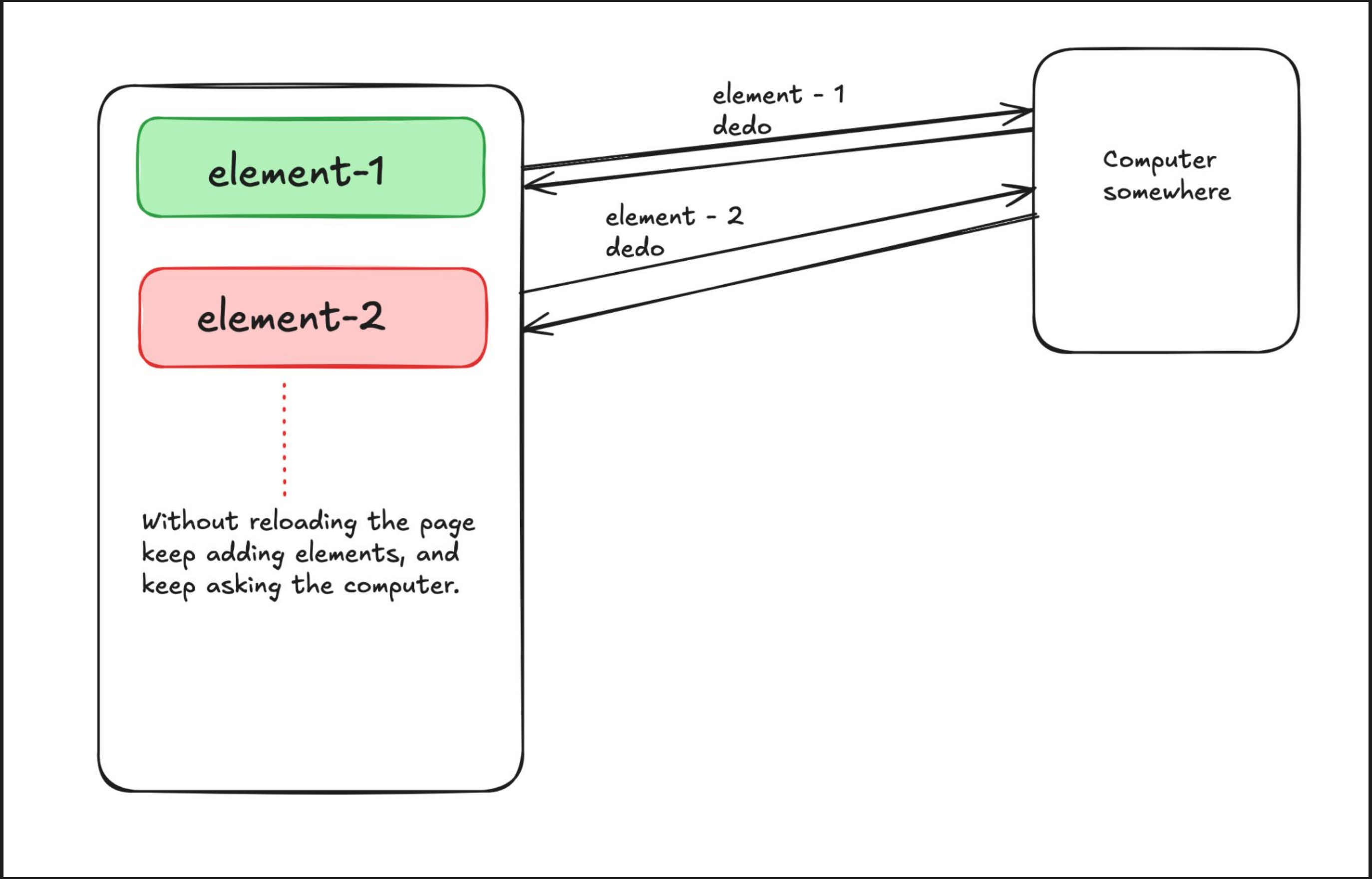
Some common events:

dblclick – Fires when an element is double-clicked.
mouseenter – Fires when the mouse enters an element.
mouseleave – Fires when the mouse leaves an element.
mouseover – Fires when the mouse enters an element or its child elements.
mouseout – Fires when the mouse leaves an element or its child elements.
keydown – Fires when a key is pressed.
keyup – Fires when a key is released.
input – Fires when the value of an input field changes.
focus – Fires when an input field is focused.
blur – Fires when an input field loses focus.
removeEventListener – Removes an attached event listener.
change – Fires when the value of an input/select changes.

AJAX

2

WHAT IS AJAX?


element-1
element-2
element - 1
dedo
element - 2
dedo
Computer
somewhere
Without reloading the page
keep adding elements, and
keep asking the computer.

WHAT IS AJAX?

- Google Maps
- Instagram/Facebook feeds - infinite scroll

Requesting data and loading it on the page is simply what Asynchronous JavaScript and XML (AJAX)

FETCH API

The `fetch()` function returns a `Promise` which is fulfilled with a `Response` object representing the server's response. You can then check the request status and extract the body of the response in various formats, including text and JSON, by calling the appropriate method on the response.

FREE PUBLIC API:

- `https://jsonplaceholder.typicode.com/todos/`
- `https://jsonplaceholder.typicode.com/todos/ 1`

FETCH API

Syntax:

```
fetch(url)
    .then(response => {
        return response.json()
    })
    .then(data => {
        console.log(data);
    }).catch(errorMessage => {
        console.log(errorMessage);
    })
```

Way-2:
This is default return of arrow function.

```
fetch(url)
    .then(res => res.json())
    .then(data => {
        console.log(data);
    }).catch(errorMessage => {
        console.log(errorMessage);
    })
```

JSON - JavaScript Object Notation.

- JSON is plain text written in JavaScript object notation - Basically it’s a string that is actually a JS object.
- JSON is used to send data between computers
- JS array of objects, or a simple object can be converted to JSON.

```
{
    name: 'Coding blocks',
    age: 10,
    description: 'Best place to learn coding',
    courses: ['CPP','Java']
}
```

This is JavaScript Object.

```
{
 "name": "Coding blocks",
 "age": 10,
 "description": "Best place to learn coding",
 "courses": [
  "CPP",
  "Java"
 ]
}
```

This is JSON.

JSON - JavaScript Object Notation.

- JSON.parse() : It will convert JSON to JS object.
- JSON.stringify() : It will convert JS to JSON.

XMLHTTPRequest?

We need to do 5 things:

1. **Create XMLHTTPRequest object**
2. **On success we need to create onload method.**
3. **On error we need to create onerror method.**
4. **To send request we use send method.**
5. **To setup we need to use open method.**

TOO MUCH WORK Right?

XMLHTTPRequest?

```
let xhr = new XMLHttpRequest();

// Request kaha karni h
xhr.open("GET", url);

// Request send krni hai jab button dab jaaye
button.onclick = () => {
    xhr.send();
}

xhr.onload = (data) => {
    // JSON: Javascript Object Notation
    // JSON data recieve hua
    data = data.currentTarget.response;

    // data is string by default, to convert it to JS object we use
    // JSON.parse()
    data = JSON.parse(data);
    addToTaskList(data);
}
// Error aane par yeh hoga!
xhr.onerror = (errorMsg) => {
    console.log(errorMsg);
}
```

AXIOS.

Most popular way to do requests.

- **Documentation: https://axios-http.com/docs/intro**

It’s not inbuilt in browser, so to use it include the link 👇 in your index.html before your Javascript code.

```
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
```

For using axios : https://axios-http.com/docs/example

Now we got the data let’s use the DOM manipulation to show the result on page.

TRY IT!

Let’s see important functions now!

- map
- filter
- reduce

IIFE
Immediately Invoked Function Expression