XMLHttpRequest

XMLHttpRequest (XHR) is a built-in JavaScript object that allows you to make HTTP requests to a server without having to reload the page. It's commonly used for fetching data from a server and updating parts of a web page without a full page reload.

```
// Create a new XMLHttpRequest object
var xhr = new XMLHttpRequest();

// Define a callback function to handle the state changes
xhr.onreadystatechange = function () {
  // Check if the request is complete (readyState === 4)
  if (xhr.readyState === 4) {
    if (xhr.status === 200) {
      console.log("Response:", xhr.responseText);
    } else {
      console.error("Request failed:", xhr.status);
    }
  } else {
    console.log("Ready state:", xhr.readyState);
  }
};

// Open a new HTTP request
xhr.open("GET", "https://jsonplaceholder.typicode.com/posts/1", true);

// Send the request
xhr.send();
```

The onreadystatechange event handler is called each time the readyState changes. The readyState property holds the status of the XMLHttpRequest. Here's what each readyState represents:

- **0: UNSENT** - The request has not been opened yet.
- **1: OPENED** - The request has been opened.
- **2: HEADERS_RECEIVED** - Headers have been received.
- **3: LOADING** - The response body is loading.
- **4: DONE** - The operation is complete.

In the example, we log the readyState to the console to see the progress of the request. When the readyState is 4, the request is complete, and we check if the status is 200 (OK). If it's 200, we log the response data. If it's not 200, we log an error message.

Fetch API

The Fetch API provides a simple interface for fetching resources (like JSON data, images, files, etc.) asynchronously across the network. It's built into modern browsers and provides a more powerful and flexible way to handle network requests compared to traditional methods like XMLHttpRequest.

Syntax

```
fetch(url, [options])
  .then((response) => {
    // Handle response
  })
  .catch((error) => {
    // Handle error
  });
```

- **url**: The URL of the resource you want to fetch.
- **options (optional)**: An optional object containing any custom settings that you want to apply to the request, such as headers, method, body, etc.

Example:

- Fetching JSON Data:

```
fetch("https://jsonplaceholder.typicode.com/posts/1")
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) => console.error("Error:", error));
```

- Posting Data:

```
fetch("https://example.com/api/resource", {
  method: "POST",
  headers: {
    "Content-Type": "application/json",
  },
  body: JSON.stringify({ key: "value" }),
})
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) => console.error("Error:", error));
```