

GitHub URL

Vercel URL

W12-P1: use xhr to display a simple text

The screenshot shows a developer environment with several panes:

- EXPLORER**: Shows a project structure with folders like 1132-1N-DEMO-43, demo, md, wub_basics_43, w06_prep_test_43, w10_supabase_43, w11_async_js_43, w01_dom_43, w02_dom_43, w02_tictactoe_43, w03_basics_43, w03_tictactoe_43, w04_basics_43, w05_basics_43, w06_prep_tests_43, w10_mid1_m3_43, w10_product_supabase_43, and w11_async_js_43. A file named sample.txt is also listed.
- index.html U JS app_43.js U**: The current files being edited. The code in app_43.js is as follows:

```
1 const xhr = new XMLHttpRequest();
2 console.log("xhr0", xhr);
3
4 xhr.open("GET", "./api/sample.txt");
5 console.log("xhr", xhr);
6
7 xhr.onreadystatechange = () => {
8   if (xhr.readyState === 4 && xhr.status === 200) {
9     const text = document.createElement('p');
10    text.textContent = xhr.responseText;
11    document.body.appendChild(text);
12  } else {
13    console.log({
14      status: xhr.status,
15      text: xhr.statusText,
16    });
17  }
18};
19
20
21 xhr.send();
```

- Browser Preview**: Displays the page "Asynchronous JS Ajax Demo" with the heading "VICTOR HSU, 213410243". The page content is:

非同步JavaScript 和XML (AJAX) 是Web 應用程式開發技術的組合，可讓Web 應用程式更快速地回應使用者互動。每當您的使用者與Web 應用程式互動時(例如當他們按一下按鈕或核取方塊時)，瀏覽器就會與遠端伺服器交換資料。資料交換可能會導致頁面重新載入並中斷使用者體驗。
- Elements**, **Console**, **Sources**, **Network**, **Performance**, **Memory**, **Application**: Developer tools for inspecting the page's DOM, network requests, and performance.
- Debugger**: Shows the state of the XMLHttpRequest object with various properties highlighted in red boxes, such as readyState, onreadystatechange, and responseText.

0609d8b victor_xu

Sun May 11 15:32:43 2025 +0800 W12-P1: use xhr to display a si

W12-P2: click a button to fetch data

The screenshot shows a development environment with several panes:

- EXPLORER**: Shows a project structure with files like `1132-1N-DEMO-43`, `demo`, `w01_dom_43`, etc., and a folder `w12_ajax_async_43` containing `2-fetch-data`, `JS app_43.js`, and `index.html`. A red box highlights the `2-fetch-data` folder.
- Javascript (JS app_43.js)**: Displays the following JavaScript code:

```
const button = document.querySelector(".btn");
button.addEventListener("click", () => {
  fetchData();
});

const fetchData = () => {
  const xhr = new XMLHttpRequest();
  console.log("xhr0", xhr);

  xhr.open("GET", "./api/sample.txt");
  console.log("xhr", xhr);

  xhr.onreadystatechange = () => {
    console.log("xhr", xhr);
    if (xhr.readyState === 4 && xhr.status === 200) {
      const text = document.createElement("p");
      text.textContent = xhr.responseText;
      document.body.appendChild(text);
    } else {
      console.log(`property: ${text}`);
      text: xhr.statusText,
    }
  };
  xhr.send();
};
```

A red box highlights the `fetchData` function.
- BROWSER**: Shows a browser window at `127.0.0.1:5500/de...` with the title "Asynchronous JS Ajax Demo". It displays the text "VICTOR HSU, 213410243" and a blue button labeled "click me". Below the button is a red box containing the following explanatory text:

非同步JavaScript 和XML (AJAX) 是Web應用程式開發技術的組合，可讓Web應用程式更快速地回應使用者互動。每當您的使用者與Web應用程式互動時(例如當他們按一下按鈕或核取方塊時)，瀏覽器就會與遠端伺服器交換資料。資料交換可能會導致頁面重新載入並中斷使用者體驗。
- NETWORK**: Shows the Network tab of the developer tools with a request for `sample.txt`. The request URL is `http://127.0.0.1:5500/demo/w12_ajax_async_43/2-fetch-data/api/sample.txt`, Method is `GET`, Status Code is `200 OK`, and Remote Address is `127.0.0.1:5500`.

See5ced victor_xu

Sun May 11 15:36:45 2025 +0800 W12-P2: click a button to fetch

W12-P3: Run w10_product_supA_43, see how it works

```
=> _supabase.from('product_43').select('*');
```

The screenshot shows a developer's environment with multiple windows open:

- Code Editor:** An IDE window displays the file `product_supabase_43.js`. The code uses Supabase to fetch products from a database table named `product_43`. A red box highlights the database query line: `let data = await _supabase.from('product_43').select('*');`
- Browser Developer Tools:** The browser's Network tab is active, showing a request for the product detail page. A red box highlights the `Headers` section, which includes the URL `https://svvqtkkzbzhmilyhbzd.supabase.co/rest/v1/product_43?select=*`, Method `GET`, Status `200 OK`, and the IP address `104.16.30.10:443`. Another red box highlights the `General` section, showing the response body: `<h1>Get Product from Supabase</h1><h2>VICTOR HSU, 213410243</h2>`.
- Product Detail Page:** The main browser window shows a product detail page for a speaker system. The title is "Get Product from Supabase" and the subtitle is "VICTOR HSU, 213410243". The page features a large image of a black speaker system, some text, and a footer.

=> check response

The screenshot shows a browser window with the URL `127.0.0.1:5500/demo/w10_product_supabase_43.js`. On the left is a code editor with the file `product_supabase_43.js`. A red box highlights the line `let { data, error } = await _supabase.from('products').select('*').single();`. On the right, the browser's developer tools Network tab is open, showing a request for `product_43s...` which returns a JSON response containing 11 products. One product is highlighted with a red box: `{ id: 10, title: "Modern Bookshelf", price: 8.99, category: "Marcos", img: "./images/product-10.jpg", remote_img: "https://svvqtkizbzhlbylnbdz.supabase.co...`. Below the Network tab, the product page is displayed with a modern bookshelf and a price of \$8.99.

=> check how many http requests being done in fetchProducts

The screenshot shows a browser window with the same URL as before. The code editor on the left has the same code, with the `fetchProducts` block highlighted by a red box. On the right, the Network tab shows 12 requests for images named `product-10.jpg` through `product-9.jpg`, each with a status of 200 and a specific file size. The product page below shows the same modern bookshelf and price.

W12-P4: Fetch person.json string and display name in the browser

=> use `JSON.parse()` to convert `responseText` to JSON array

The screenshot shows a development setup with two main panes. On the left is a code editor (VS Code) displaying `JS app_43.js` with the following code:

```

1 const button = document.querySelector(".btn");
2 const url = "./api/person.json";
3 button.addEventListener("click", () => {
4   fetchData(url);
5 });
6
7 const fetchData = (url) => {
8   const xhr = new XMLHttpRequest();
9   console.log("xhr8", xhr);
10
11   xhr.open("GET", url);
12   console.log("xhr", xhr);
13
14   xhr.onreadystatechange = () => {
15     console.log("xhr", xhr);
16     if (xhr.readyState === 4 && xhr.status === 200) {
17       const data = JSON.parse(xhr.responseText);
18       console.log("data", data);
19       console.log("data in string", JSON.stringify(data));
20       // const displayData = data.map((item) => `<p>${item.id}: ${item.name}</p>`);
21       const element = document.createElement("div");
22       // element.innerHTML = displayData;
23       element.innerHTML = xhr.responseText;
24       document.body.appendChild(element);
25     } else {
26       console.log({
27         status: xhr.status,
28         text: xhr.statusText,
29       });
23     }
24   };
25
26   xhr.send();
27 };
28
29
30
31
32
33
34 };
35

```

On the right is a browser window showing the result of the code execution. The title is "Asynchronous JS Ajax Demo" and the content area contains a button labeled "click me". Below the button, the browser displays the JSON array received from the API:

```

[{"id": 1, "name": "John Doe"}, {"id": 2, "name": "Jane Smith"}, {"id": 3, "name": "Alice Johnson"}, {"id": 4, "name": "Bob Brown"}, {"id": 5, "name": "Charlie Black"}]

```

The browser's developer tools are open, showing the network tab with a request to `http://127.0.0.1:5500/demo/w12_ajax_asy...`. The response tab shows the JSON data received:

```

responseText: "[{"id": 1, "name": "John Doe"}, {"id": 2, "name": "Jane Smith"}, {"id": 3, "name": "Alice Johnson"}, {"id": 4, "name": "Bob Brown"}, {"id": 5, "name": "Charlie Black"}]"

```

The developer tools also show the `data` variable in the global scope, which is an array of objects representing the fetched data.

=> extract name from data and show it in the browser

The screenshot shows a developer environment with two main windows: VS Code on the left and a browser window on the right.

VS Code Explorer:

- Project structure: 1132-IN-DEMO-43
- demo folder contains: md, w11_async_js_43, w01_dom_43, w02_dom_43, w02_tictactoe_43, w03_basics_43, w03_tictactoe_43, w04_basics_43, w05_basics_43, w06_prep_tests_43, w10_mid1_m3_43, w10_product_supra..., w11_async_js_43, w12_ajax_asy..., index.html, JS app_43.js, sample.txt.
- 3-fetch-json folder contains: api, person.json, sample.txt.
- index.html file is open in the editor.

Code Editor (JS app_43.js):

```
const button = document.querySelector(".btn");
const url = "./api/person.json";
button.addEventListener("click", () => {
  fetchData(url);
});

const fetchData = (url) => {
  const xhr = new XMLHttpRequest();
  console.log("xhr0", xhr);

  xhr.open("GET", url);
  console.log("xhr", xhr);

  xhr.onreadystatechange = () => {
    console.log("xhr", xhr);
    if (xhr.readyState === 4 && xhr.status === 200) {
      const data = JSON.parse(xhr.responseText);
      console.log("data", data);
      // console.log('data in string', JSON.stringify(data));
      const displayData = data.map((item) => `<p>${item.name}</p>`);
      const element = document.createElement("div");
      element.innerHTML = displayData;
      // element.innerHTML = xhr.responseText;
      document.body.appendChild(element);
    } else {
      console.log({
        status: xhr.status,
        text: xhr.statusText,
      });
    }
  };
  xhr.send();
};
```

Browser Window:

Asynchronous JS Ajax Demo

VICTOR HSU, 213410243

click me

Victor Hsu
213410243
Alice Johnson
Bob Brown
Charlie Black

The browser window displays the output of the JavaScript code. It shows a list of names: Victor Hsu, 213410243, Alice Johnson, Bob Brown, and Charlie Black. The names are displayed in separate paragraphs within a div.

Console Tab (Browser DevTools):

```
xhr0 XMLHttpRequest {onreadystatechange: null, readyState: 0, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, _}
xhr XMLHttpRequest {onreadystatechange: null, readyState: 1, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, _}
xhr XMLHttpRequest {onreadystatechange: null, readyState: 2, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, onreadystatechange: f, _}
> {status: 200, text: 'OK'} app.43.js:26
xhr XMLHttpRequest {onreadystatechange: 3, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, onreadystatechange: f, _}
> {status: 200, text: 'OK'} app.43.js:26
xhr XMLHttpRequest {readyState: 4, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, onreadystatechange: f, _}
data [5] [Object object]
0: {id: 1, name: 'Victor Hsu'}
1: {id: 2, name: '213410243'}
2: {id: 3, name: 'Alice Johnson'}
3: {id: 4, name: 'Bob Brown'}
4: {id: 5, name: 'Charlie Black'}
length: 5
[[Prototype]: Array(0)]
```

=> check the Network, http response

The screenshot shows the Network tab in the Chrome DevTools interface. The 'Network' tab is selected, highlighted with a red box. Below it, the 'Response' tab is also highlighted with a red box. The request 'person.json' is listed under the 'Name' column. The response body is displayed in a code editor-like view, showing a JSON array of five objects. Each object has an 'id' and a 'name' field. The entire JSON structure and its individual elements are highlighted with a red box.

Name	X	Headers	Preview	Response	Initiator	Timing	Cookies
person.json				<pre>1 [2 { 3 "id": 1, 4 "name": "Victor Hsu" 5 }, 6 { 7 "id": 2, 8 "name": "213410243" 9 }, 10 { 11 "id": 3, 12 "name": "Alice Johnson" 13 }, 14 { 15 "id": 4, 16 "name": "Bob Brown" 17 }, 18 { 19 "id": 5, 20 "name": "Charlie Black"</pre>			

357d508 victor_xu

Sun May 11 16:03:18 2025 +0800 W12-P4: Fetch person.json strin

W12-logs: git logs of W12

The screenshot shows a GitHub repository interface for the branch 'master'. The main area displays the commit history for May 11, 2025, with four commits highlighted by a red box:

- W12-P4: Fetch person.json string and display name in the browser** (357d508) - vic0627 committed 4 minutes ago
- W12-P3: Run w10_product_supra_43, see how it works** (400b44b) - vic0627 committed 4 minutes ago
- W12-P2: click a button to fetch data** (5ee5ced) - vic0627 committed 33 minutes ago
- W12-P1: use xhr to display a simple text** (0609d8b) - vic0627 committed 37 minutes ago

The sidebar on the right shows a file tree for the repository, including files like .nx, css, demo, md, w01_dom_43, w02_43, w03_tictactoe, w04_basics, w05_basics, w06_prep, w07_supra, w08_supra, w09_supra, w10_supra_43, w11_async_ajax, w11_43, w11_43.pot, w11-logs, w11-p1-1, w11-p1-2, w11-p2, w11-p2.pot, w11-p3, w11-p3.pot, w11-p4, w11-p4.pot, w11-p5-1, w11-p5-2, w12_43, w12-p1, w12-p2, w12-p3-1, and w12-p3-2.