

# Github-repo url

[https://github.com/chen945/1102\\_javascript\\_76](https://github.com/chen945/1102_javascript_76)

## w13-p1: use xhr object to get sample.txt, and show it on webpage

The screenshot displays a web development environment with three main components: a code editor on the left, a web browser in the center, and a Chrome DevTools console on the right.

**Code Editor (Left):** The file `index.html` is open, showing JavaScript code that uses the `XMLHttpRequest` (XHR) object to fetch data from `./api/sample.txt`. The code includes comments in Chinese and a `console.log` statement to verify the response. The `onreadystatechange` event listener is highlighted with a red box.

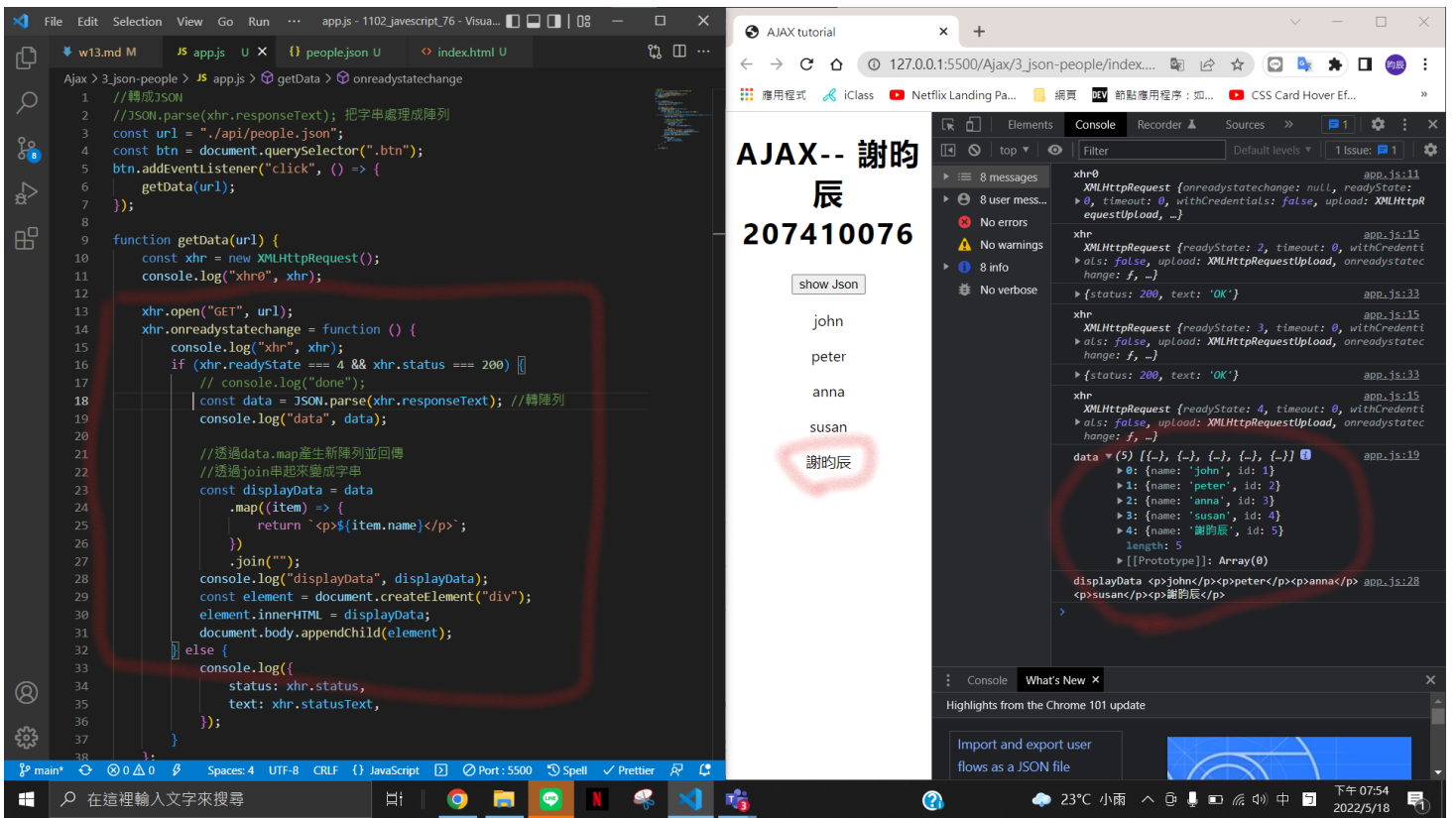
```
1 //從sample.txt 撈資料
2 const xhr = new XMLHttpRequest();
3 console.log("xhr0", xhr);
4
5 xhr.open("GET", "./api/sample.txt");
6 xhr.onreadystatechange = function () {
7   console.log("xhr", xhr);
8   if (xhr.readyState === 4 && xhr.status === 200)
9     // console.log("done");
10    const text = document.createElement("p");
11    text.textContent = xhr.responseText;
12    document.body.appendChild(text);
13  } else {
14    console.log({
15      status: xhr.status,
16      text: xhr.statusText,
17    });
18  }
19 };
20
21 xhr.send("");
22 console.log("hello");
23
```

**Web Browser (Center):** The browser shows the page `AJAX-- 謝昀辰 207410076`. Below the title, the text `謝昀辰-- 207410076 Lorem, ipsum dolor sit amet consectetur adipisicing elit. Voluptate quia ipsa sequi animi reprehenderit atque voluptatibus non eaque repudiandae alias molestias cumque, dolore, unde et nesciunt esse tenetur suscipit dicta.` is displayed and circled in red.

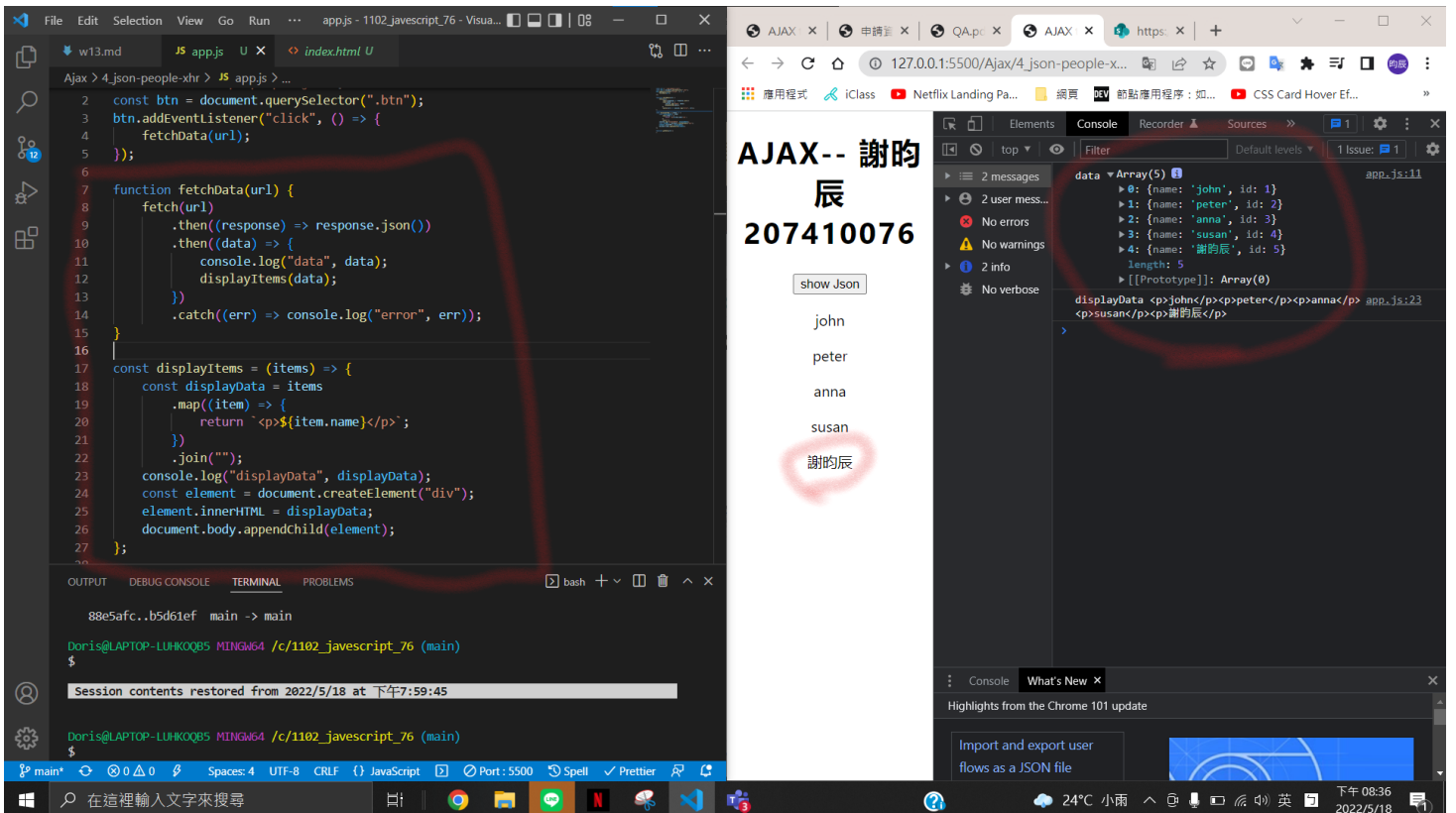
**Chrome DevTools Console (Right):** The console shows the XHR request details. The `onreadystatechange` event is highlighted with a red box, showing the `readyState` as `4` and the `status` as `200`. The `responseText` is the same text seen on the webpage.

```
{
  status: 200,
  text: "OK"
}
XMLHttpRequest {readyState: 3, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, onreadystatechange: f, ...}
{status: 200, text: "OK"}
XMLHttpRequest {readyState: 4, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, onreadystatechange: f, ...}
onreadystatechange: f ()
onabort: null
onerror: null
onload: null
onloadend: null
onloadstart: null
onprogress: null
onreadystatechange: f ()
onreadystatechange: f ()
response: "謝昀辰-- 207410076 Lorem, ipsum dolor sit amet consectetur adipisicing elit. Voluptate quia ipsa sequi animi reprehenderit atque voluptatibus non eaque repudiandae alias molestias cumque, dolore, unde et nesciunt esse tenetur suscipit dicta."
responseText: "謝昀辰-- 207410076 Lorem, ipsum dolor sit amet consectetur adipisicing elit. Voluptate quia ipsa sequi animi reprehenderit atque voluptatibus non eaque repudiandae alias molestias cumque, dolore, unde et nesciunt esse tenetur suscipit dicta."
responseXML: null
status: 200
statusText: "OK"
timeout: 0
upload: XMLHttpRequestUpload {onloadstart: null, onwithCredentials: false}
[[Prototype]]: XMLHttpRequest
```

## w13-p2: use xhr object to get people.json, and show it on webpage



w13-p3: use fetch api to get people.json, and show it on webpage



## w13-p4: use async/await to get people.json, and show it on webpage

The screenshot shows a development environment with VS Code on the left and a web browser on the right. The VS Code editor displays the file explorer on the left with the following structure:

- 1102\_JAVASCRIPT\_76
  - Ajax
    - 1\_simple-text
    - 2\_add-btn
    - 3\_json-people...
    - 4\_json-people...
    - 5\_json-people...
    - api
  - JS app.js
  - index.html
  - 6\_midterm\_p2...
  - array\_demo\_76
  - Async\_Javascript
  - exam
  - guess-starter-76
  - md
  - HW1-guess numbers
  - midthem
  - w01
  - w02
  - w03
  - w04
  - w05
  - w08
  - w12
  - w13
  - p1.png

The main editor shows the code in `app.js`:

```
1 const url = './api/people.json';
2 const btn = document.querySelector('.btn');
3 btn.addEventListener('click', () => {
4   fetchDataAsync(url);
5 });
6
7 const fetchDataAsync = async (url) => {
8   try {
9     const response = await fetch(url);
10    const data = await response.json();
11    console.log("data", data);
12    displayItems(data);
13  } catch (err) {
14    console.log("error", err);
15  }
16 };
17
18 > function fetchData(url) { ...
19 }
20
21 > function displayItems = (items) => { ...
22 }
23
24 > function getData(url) { ...
25 }
26
27 >
28 >
29 >
30 >
31 >
32 >
33 >
34 >
35 >
36 >
37 >
38 >
39 >
40 >
41 >
42 >
43 >
44 >
45 >
46 >
47 >
48 >
49 >
50 >
51 >
52 >
53 >
54 >
55 >
56 >
57 >
58 >
59 >
60 >
61 >
62 >
63 >
64 >
65 >
66 >
67 >
68 >
69 >
```

The web browser shows the page `127.0.0.1:5500/Ajax/5_json-people-async-await/in...` with the title `AJAX-- 謝昀辰 207410076`. A button labeled `show Json by async/await` is present. Below the button, the names `john`, `peter`, `anna`, and `susan` are displayed. The console shows the following log:

```
data
0: {name: 'john', id: 1}
1: {name: 'peter', id: 2}
2: {name: 'anna', id: 3}
3: {name: 'susan', id: 4}
4: {name: '謝昀辰', id: 5}
length: 5
[[Prototype]]: Array(0)
```

## w13-p5:自主練習

The screenshot shows a development environment with VS Code on the left and a web browser on the right. The VS Code editor displays the file explorer on the left with the following structure:

- 1102\_JAVASCRIPT\_76
  - Ajax
    - 1\_simple-text
    - 2\_add-btn
    - 3\_json-people...
    - 4\_json-people...
    - 5\_json-people...
    - api
  - JS app.js
  - index.html
  - 6\_midterm\_p2...
  - array\_demo\_76
  - Async\_Javascript
  - exam
  - guess-starter-76
  - md
  - HW1-guess numbers
  - midthem
  - w01
  - w02
  - w03
  - w04
  - w05
  - w08
  - w12
  - w13
  - p1.png

The main editor shows the code in `app.js`:

```
1 const url = './api/dataset.json';
2 const btn = document.querySelector('.btn');
3 let data = [];
4
5 btn.addEventListener('click', () => {
6   fetchDataSet(url);
7 });
8
9 const fetchDataSet = async (url) => {
10  try {
11    const response = await fetch(url);
12    const data = await response.json();
13    console.log("data", data);
14    updateDOM(data);
15  } catch (err) {
16    console.log("error", err);
17  }
18 };
19
20 const fetchDataAsync = async (url) => {
21  try { ...
22  } catch (err) {
23    console.log("error", err);
24  }
25 };
26
27 >
28 >
29 >
30 >
31 >
32 >
33 >
34 >
35 >
36 >
37 >
38 >
39 >
40 >
41 >
42 >
43 >
44 >
45 >
46 >
47 >
48 >
49 >
50 >
51 >
52 >
53 >
54 >
55 >
56 >
57 >
58 >
59 >
60 >
61 >
62 >
63 >
64 >
65 >
66 >
67 >
68 >
69 >
```

The web browser shows the page `127.0.0.1:5500/Ajax/6_midterm_p2_76/index.html` with the title `AJAX-- 謝昀辰 207410076`. A button labeled `show Json` is present. Below the button, the text `Person Wealth` is displayed. Below this, the following table is shown:

Ellen Cruz	\$561,575,400.00
Raul Velasco	\$41,338,300.00
Konsta Tuominen	\$112,175,400.00
謝昀辰	\$123,754,000.00

The console shows the following log:

```
data
0: {name: 'Ellen Cruz', money: 561575400}
1: {name: 'Raul Velasco', money: 41338300}
2: {name: 'Konsta Tuominen', money: 112175400}
3: {name: '謝昀辰', money: 123754000}
length: 4
[[Prototype]]: Array(0)
```

# Log

```
Doris@LAPTOP-LUHKQBS MINGW64 /c/1102_javascript_76 (main)
$ git log --pretty=format:"%h%x09%an%x09%ad%x09%s" --after="2022-05-17"
389a274 chen945 Wed May 18 21:38:54 2022 +0800 w13-p5:自主練習
a9c796e chen945 Wed May 18 20:55:50 2022 +0800 w13-p4: use async/await to get people.json, and show it on webpage
e2de7da chen945 Wed May 18 20:38:02 2022 +0800 w13-p3: use fetch api to get people.json, and show it on webpage
b5d61ef chen945 Wed May 18 19:56:29 2022 +0800 w13-p2: use xhr object to get people.json, and show it on webpage
88e5afc chen945 Wed May 18 19:06:53 2022 +0800 w13-p1: use xhr object to get sample.txt, and show it on webpage
```

```
$ git log --pretty=format:"%h%x09%an%x09%ad%x09%s" --after="2022-05-17"
389a274 chen945 Wed May 18 21:38:54 2022 +0800 w13-p5:自主練習
a9c796e chen945 Wed May 18 20:55:50 2022 +0800 w13-p4: use async/await to get people.json, and show it on webpage
e2de7da chen945 Wed May 18 20:38:02 2022 +0800 w13-p3: use fetch api to get people.json, and show it on webpage
b5d61ef chen945 Wed May 18 19:56:29 2022 +0800 w13-p2: use xhr object to get people.json, and show it on webpage
88e5afc chen945 Wed May 18 19:06:53 2022 +0800 w13-p1: use xhr object to get sample.txt, and show it on webpage
```