

My Github URL

W13-P1: Use fetch API to replace xhr approach

```
demo > w13_fetch_async_36 > 4-fetch-api > JS app_36.js > ...
6 btn.addEventListener('click', () => {
10 })
11
12 > const getDataByFetchAPI = (url) => { ...
25 }
26
27 const getDataByFetchAPI2 = (url) => {
28   fetch(url)
29   .then((response)=> response.json())
30   .then((data)=> {
31     console.log('data', data)
32     displayItems(data)
33     // return data
34   })
35   .catch((err) => console.log(err))
36 }
37
38 const displayItems = (persons) => {
39   const displayData = persons
40   .map((person) => {
41     return `
42     <p>${person.name}</p>
43   })
44   .join('')
45   const element = document.createElement('div')
46   element.innerHTML = displayData
47   document.body.appendChild(element)
48 }
49
50
51 > const getDataByXHR = () => { ...
83 }
84
85
```

Asynchronous JS - Ajax Demo

何柏鑫, 213410136

click me

何柏鑫
213410136
Andy
Betty

Console

data (4) [{"id": 1, "name": "何柏鑫"}, {"id": 2, "name": "213410136"}, {"id": 3, "name": "Andy"}, {"id": 4, "name": "Betty"}]

W13-P2: Use Async Await to replace fetch API approach

```
1 const btn = document.querySelector('.btn')
2 const url = './api/person.json'
3
4 btn.addEventListener('click', () => {
5   getDataAsyncAwait(url)
6 })
7
8 const getDataAsyncAwait = async (url) => {
9   try{
10     const response = await fetch(url)
11     const data = await response.json()
12     console.log('data', data)
13     displayItems(data)
14   }catch(err){
15     console.log(err)
16   }
17 }
18
19 > const getDataByFetchAPI = (url) => { ...
32 }
33
34 const getDataByFetchAPI2 = (url) => {
35   fetch(url)
36   .then((response)=> response.json())
37   .then((data)=> {
38     console.log('data', data)
39     displayItems(data)
40     // return data
41   })
42   .catch((err) => console.log(err))
43 }
44
```

Asynchronous JS - Async Await Demo

何柏鑫, 213410136

click me

何柏鑫
213410136
Andy
Betty

Console

data (4) [{"id": 1, "name": "何柏鑫"}, {"id": 2, "name": "213410136"}, {"id": 3, "name": "Andy"}, {"id": 4, "name": "Betty"}]

W13-P3: Get meals about cheese from TheMealDB

The screenshot displays the development environment for W13-P3. On the left, the VS Code editor shows the `app_36.js` file with the following code:

```

1 const btn = document.querySelector('.btn')
2 const url = 'https://www.themealdb.com/api/json/v1/1/search.php?s=cheese'
3
4 btn.addEventListener('click', () => {
5   getDataAsyncAwait(url)
6 })
7
8 const getDataAsyncAwait = async (url) => {
9   try {
10    const response = await fetch(url)
11    const data = await response.json()
12    console.log('data.meals', data.meals)
13    displayItems(data.meals)
14  } catch (err) {
15    console.log(err)
16  }
17 }
18
19 const displayItems = (data) => {
20   const displayData = data
21   .map((item) => {
22     return `
23     <p>${item.strMeal}</p>
24   `)
25   .join('')
26   const element = document.createElement('div')
27   element.innerHTML = displayData
28   document.body.appendChild(element)
29 }

```

On the right, the web browser shows the 'The MealDB API Demo' page. A 'click me' button is visible. Below it, a list of meals is displayed, including 'Cream Cheese Tart', 'New York cheesecake', 'Three-cheese souffles', 'Honey Yogurt Cheesecake', 'Peanut Butter Cheesecake', 'Chicken Fajita Mac and Cheese', 'Grilled Mac and Cheese Sandwich', and 'Fruit and Cream Cheese Breakfast Pastries'. The 'Cream Cheese Tart' is selected, showing its details in the right sidebar, including ingredients, instructions, and a thumbnail image.

0634222 vincent560

Thu May 15 20:26:02 2025 +0800 p3

W13-P4: Get Products from local json and from API

=> Get products from local json

The screenshot displays the development environment for W13-P4. On the left, the VS Code editor shows the `product_localjson_36.js` file with the following code:

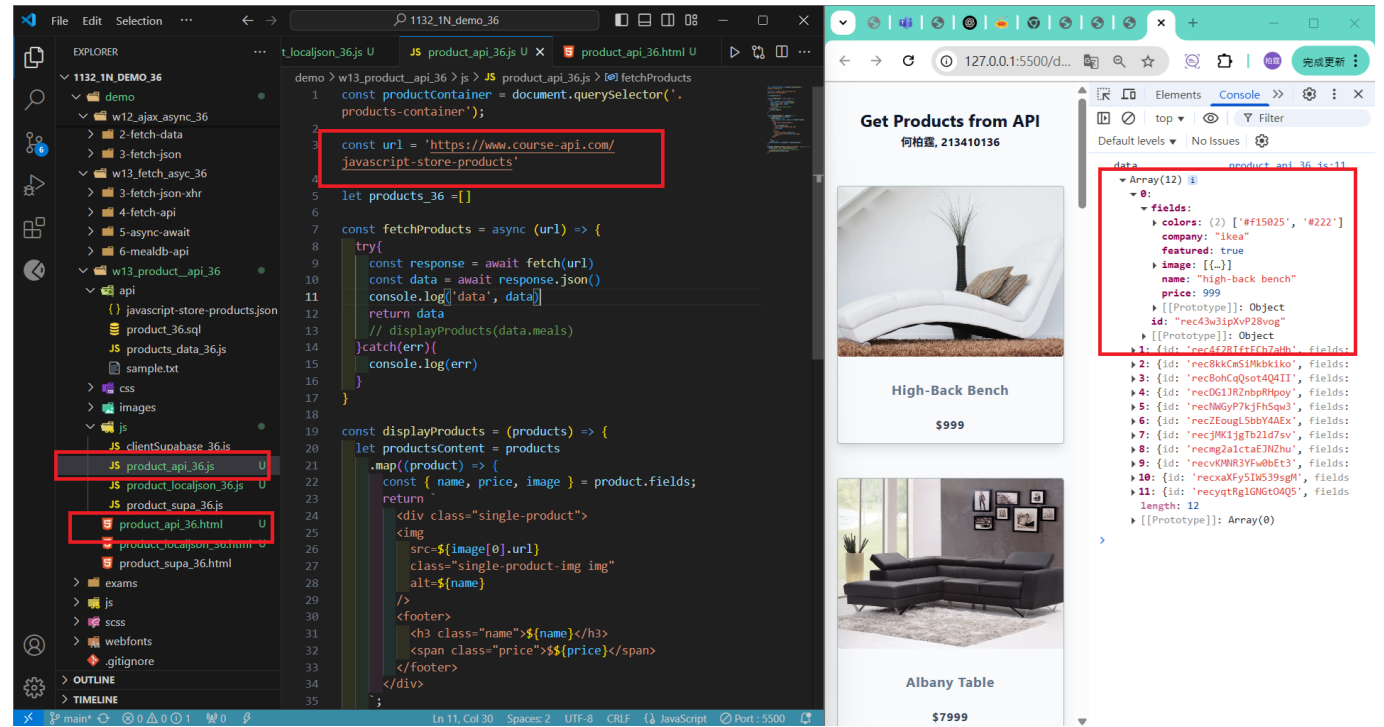
```

1 const productContainer = document.
2   querySelector('.products-container')
3
4 const url = './api/
5   javascript-store-products.json'
6
7 let products_36 = []
8
9 const fetchProducts = async (url) => {
10   try {
11     const response = await fetch(url)
12     const data = await response.json()
13     console.log('data', data)
14     return data
15   } catch (err) {
16     console.log(err)
17   }
18 }
19
20 const displayProducts = (products) => {
21   let productsContent = products
22   .map((product) => {
23     const { name, price, image } =
24       product.fields
25     return `
26     <div class="single-product">
27       <img
28         src=${image[0].url}
29         class="single-product-img img"
30         alt=${name}
31       />
32       <div>
33         <h3 class="name">${name}</h3>
34         <span class="price">${price}</span>
35       </div>
36     </div>
37   `)
38   productContainer.innerHTML =
39     productsContent
40 }
41
42 document.addEventListener(
43   'DOMContentLoaded', async () => {
44     products_36 = await fetchProducts(
45       url)
46     // console.log('products_36',
47     // products_36)
48     displayProducts(products_36)
49   })

```

On the right, the web browser shows the 'Get Products from Local JSON' page. A 'click me' button is visible. Below it, a list of products is displayed, including 'High-Back Bench' and 'Albany Table'. The 'High-Back Bench' is selected, showing its details in the right sidebar, including its price, image, and fields.

=> Get products from API



f30b65a vincent560 Sun May 18 23:16:53 2025 +0800 p4

W13-logs: git logs of W13

Commits		
main		
All users		
All time		
Commits on May 18, 2025		
p4		
vincent560 committed 1 minute ago · 1 / 1		
f30b65a		
Commits on May 15, 2025		
p3		
vincent560 committed 3 days ago · 1 / 1		
0634222		
p1p2		
vincent560 committed 3 days ago · 1 / 1		
d1ff0d3		