

React

Axios vs Fetch





Why Do We Need HTTP Request Tools?

Handling HTTP requests can be complex, but tools like **Axios and Fetch** **simplify the process** with abstractions that streamline error handling, response parsing, and request configuration.



**They help address common problems
such as:**

- Boilerplate Code
- Error Handling
- Interceptors



- **What is axios**

Axios is a **promise-based library** for **making HTTP requests**, offering more features and convenience than the native Fetch API.



Axios - Example

```

// Making a GET request using Axios
axios.get('https://api.example.com/data')
  .then(response => console.log(response.data))
  .catch(error => console.error('There was a
problem with the axios request:', error));

```




- **What is Fetch**

The Fetch API is a **promise-based JavaScript method** for making HTTP requests, offering a **simpler alternative** to older methods like XMLHttpRequest.



Fetch – Example

```
● ● ●  
  
// Making a GET request using Fetch  
  
fetch('https://api.example.com/data')  
  
  .then(response => {  
    if (!response.ok) {  
      throw new Error('Network response was not ok');  
    }  
    return response.json();  
  })  
  
  .then(data => console.log(data))  
  .catch(error => console.error('There was a problem with the  
fetch operation:', error));
```



- **Key Differences**

1. Default Handling of JSON

Fetch

Requires manual conversion of response data to JSON.

```
fetch('https://api.example.com/data')  
  .then(response => response.json()) // Manual conversion  
  .then(data => console.log(data));
```

Axios

Automatically parses JSON responses.

```
axios.get('https://api.example.com/data')  
  .then(response => console.log(response.data)); // Automatic conversion
```




2. Error Handling

Fetch

Only rejects a promise for network errors, not for HTTP errors (e.g., 404 or 500 status codes).

```
fetch('https://api.example.com/data')
  .then(response => {
    if (!response.ok) {
      throw new Error('Network response was not ok');
    }
    return response.json();
  })
  .catch(error => console.error('Fetch error:', error));
```

Axios

Rejects a promise for both network errors and HTTP errors.

```
axios.get('https://api.example.com/data')
  .catch(error => console.error('Axios error:', error));
```



3. Request Configuration

Fetch

Requires manual configuration of options like headers

```
fetch('https://api.example.com/data', {  
  method: 'POST',  
  headers:  
    { 'Content-Type': 'application/json' },  
  body: JSON.stringify({ key: 'value' })  
});
```



Axios

Provides a more concise and readable syntax for

```
axios.post('https://api.example.com/data', { key: 'value' }, {  
  headers: {  
    'Content-Type': 'application/json'  
  }  
});
```


**Was it
helpful?
follow for more!**



Muhammad Zubair
@iamzubairarain



Like



Comment



Share



Save

Did you know? I'm a freelancer too! 📁 Visit my Upwork profile:
[<https://www.upwork.com/freelancers/~014ffd655fc0791835>]