JavaScript

Mastering

JavaScript Fetch API

What is Fetch?

Fetch provides a powerful, logical way to fetch resources asynchronously.

Basic Fetch Request

The most simple use of Fetch is a GET request, which can be done like this:

```
fetch('https://api.example.com/data');
```

Handling Responses

Fetch returns a Promise that resolves to the Response of a request, whether it is successful or not.

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

Checking Successful Fetch

Often, you will want to check if the request was successful. Here's how to do it:

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

Handling JSON Data

To extract the JSON body content from the response, we use the json() method:

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

Error Handling

If a network error occurs, the catch() is triggered.

```
fetch('https://api.example.com/data')
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) =>
    console.error('There has been a problem with your
                   fetch operation:', error),
  );
```

Fetch POST Request

Fetch is not just for GET requests. You can use it to send other types of requests, like POST.

```
fetch('https://api.example.com/data', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ id: '200' }),
});
```