An API, or Application Programming Interface, is a set of rules and protocols that allows different software applications to communicate with each other. It defines the methods and data formats that applications can use to request and exchange information. APIs are commonly used to enable the integration of different systems, allowing them to work together and share data.

In the context of web development, an API typically refers to a web API, which is a collection of endpoints exposed by a web server to enable communication between a client (e.g., a web browser or a mobile app) and a server. Web APIs are often used to request and exchange data, perform actions, or access services over the internet.

To send a request to a web API in JavaScript, you can use the XMLHttpRequest object or the more modern Fetch API. Here's how you can use both methods to make a simple GET request to an API:

Using the Fetch API (Modern Approach):

```javascript
fetch('https://api.example.com/data')
  .then(response => {
    if (!response.ok) {
      throw new Error('Network response was not ok');
    }
    return response.json(); // Parse the response body as JSON
  })
  .then(data => {
    // Work with the data received from the API
    console.log(data);
  })
  .catch(error => {
    console.error('There was a problem with the fetch operation:', error);
  });
```

The Fetch API provides a more modern and flexible way to work with APIs and is widely used in modern web development.

Using the XMLHttpRequest (Older Approach):

```javascript
var xhr = new XMLHttpRequest();
xhr.open('GET', 'https://api.example.com/data', true);
xhr.onload = function() {
  if (xhr.status >= 200 && xhr.status < 300) {
    var data = JSON.parse(xhr.responseText);
    // Work with the data received from the API
    console.log(data);
  } else {
    console.error('Request failed with status', xhr.status);
  }
};
xhr.send();
```

The XMLHttpRequest approach is older and less commonly used in modern web development but is still viable.

In both examples, you specify the URL of the API you want to request, handle the response, and work with the data received from the API once the request is complete. You can customize the request method, headers, and other parameters as needed, depending on the specific API you're working with.