Fetch versus Axios: When and Where

Fetch, the replacement for XmlHTTPRequest object (xhr for short) has been around long enough for me to consider it well baked into the platform and safe to use.

As a quick refresher, this script will fetch the latest 10 posts from my blog in JSON format that we can then feed into a framework or templating engine.

We're using async/await to make the code easier to read.

```
const url = "https://publishing-project.rivendellweb.net/wp-json/wp/v2/pos
const getData = async (url) => {
  try {
    const response = await fetch(url);
    const json = await response.json();
    console.log(json);
  } catch (error) {
    console.log(error);
  }
};
getData(url);
```

While this is widely supported in browsers this is not the case with Node. There is no built-in equivalent so we have to go the module route.

The easiest way to do it is to install node-fetch which provides a syntax identical to native fetch.

After you install it in your project use async/await with the same syntax as the native fetch example.

```
const fetch = require("node-fetch");

const urlToFetch = "https://publishing-project.rivendellweb.net/wp-json/wp
const getData = async (urlToFetch) => {
```

```
  try {
    const response = await fetch(urlToFetch);
    const json = await response.json();
    console.log(json);
  } catch (error) {
    console.log(error);
  }
};
getData(urlToFetch);
```

There are some drawbackks to the fetch methods of retrieving data. The biggest one is that there is now way to cancela fetch request already in process.

There are third party libraries that work with and enchance the native fetch functionality, I've chosen to work with Axios as a replacement for native fetch.

The two differences between the `axios` version, below, and the `node-fetch` from the previous examples:

- Axios uses `get` instead of `fetch`
- The data payload appears in `response.data` instead of `response.json`

Otherwise the codde is the same.

```
const axios = require("axios");
const urlToFetch = "https://publishing-project.rivendellweb.net/wp-json/wp

const getData = async urlToFetch => {
  try {
    const response = await axios.get(urlToFetch);
    const data = response.data;
    console.log(data);
  } catch (error) {
    console.log(error);
  }
};

getData(url);
```

So, which one do we use?

It depends on who are your target users, what browsers they use and whether you're working client-side only or with Node.

But as with everything else on the web, test for performance and test for support in your target platforms; if it ain't broken, don't fix it.