Module 4-7

Async Communication in JavaScript

Asynchronous Communication

- When our JS application calls an external app to request data, we will not wait until the response returns before we finish rendering the page.
  - The remote resource will get back to us at some time in the future.

- The other parts of the JS app will continue to run or load while we wait.

- This style of handling requests for data is known as **asynchronous**.

External Resource?

- Thus far, we have been storing datas in arrays in the form of JSON objects, but this is not a flexible approach.
  - Today we will take this a step further and read from an external json data file.

- In modern application architecture, this data is obtained from an API.
  - Sound familiar :)? (Module 4-1)

- We will try to have our JS app obtain data from a file and preview how to do likewise from a real one.

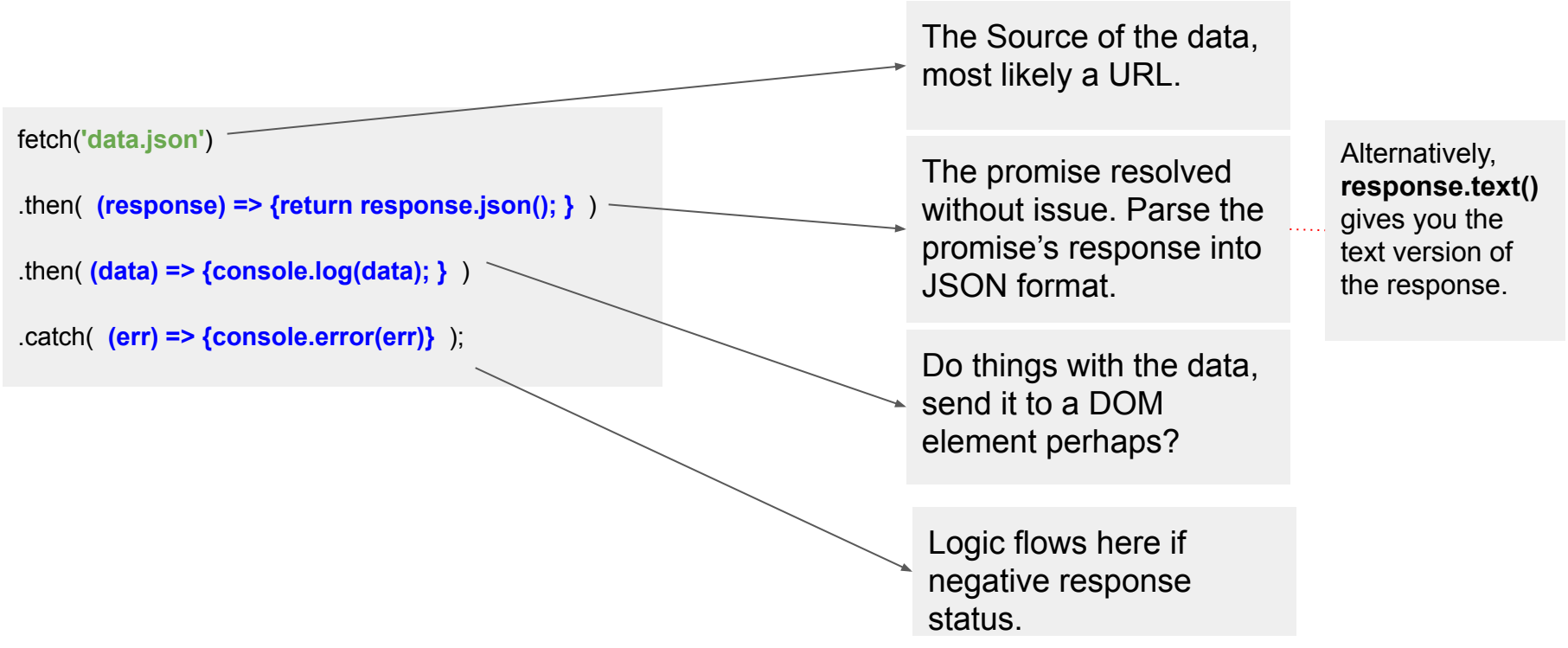
The Mighty Fetch Function

- The fetch function is used to obtain data from some other place in the network or on the internet (hint hint: API’s maybe?).

- The fetch function does not actually return the online response itself, but a **Promise object**, which represents the future success or failure of attempting to access the given resource.

Fetch Promise Chaining

We will be following a pattern called promise chaining:



Fetch Promise Chaining

The flow of this process when evaluated in the context of other JS code is not trivial. Consider this:

```
console.log("Before the call");

    fetch("data.json")
    .then(response => {
        console.log("Got the response");
        return response.text();
    })
    .then(data => {
        console.log("Got the data");
    });

 console.log("After the call");
```

Console

> Before the call
> After the call
> Got the response
> Got the data

We should not interpret these events as linear, remember that fetch is obtaining a resource from across the network, you will more likely see the statement “After the Call” execute first.

Let’s implement the fetch

In order to test fetch through our JS application, we must use the Live Server functionality of VS Code.

…. and now, the Puppies API