READING STREAMS

Fetch returns a readable Stream object

But how do we read this stream?

There are 2 important objects we need to deal with in order to read streams:

ReadableStream

ReadableStreamDefaultReader

READING STREAMS

Let’s see an example

Calling getReader() on a ReadableStream object returns another object – the actual reader

READING STREAMS

ReadableStreamDefaultReader

Why is this reader useful?

It allows us to read the Stream. We can read data in chunks, where a chunk is an array of bytes

Once we have a ReadableStreamDefaultReader object we can access the data using the read() method

READING STREAMS

So with the reader, we can see and access all the data

But, its meaningless

So, in order to transform bytes into characters (forming part of the utf-8 character set), we need to use an Encoding API

READING STREAMS

One such encoder is the TextDecoder

This is given to us by browsers

Let’s see it in action

The fetch() API returns us a readable Stream

We can access the Stream by accessing the body of the response object

To create our reader object, we can execute getReader()

To read the data, we can then execute the read() function which returns 2 values: value & done

Combining this with a decoder, we have everything we need to use chunks of data as they arrive