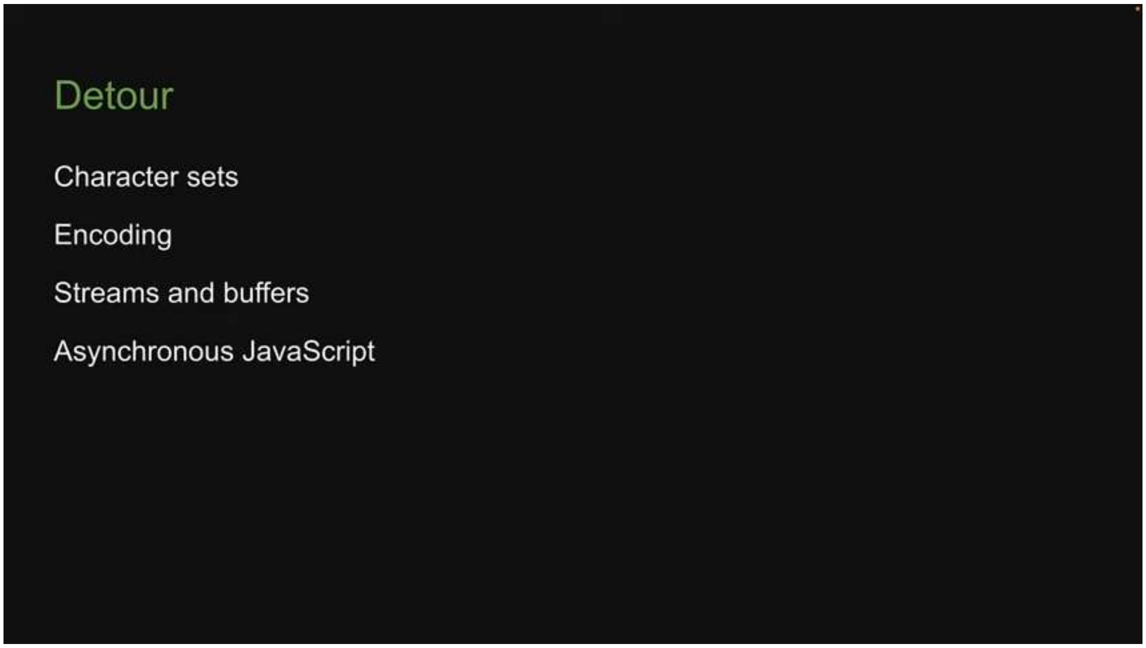
Character Set, Encoding & Streams & Buffers

05 April 2024 15:51



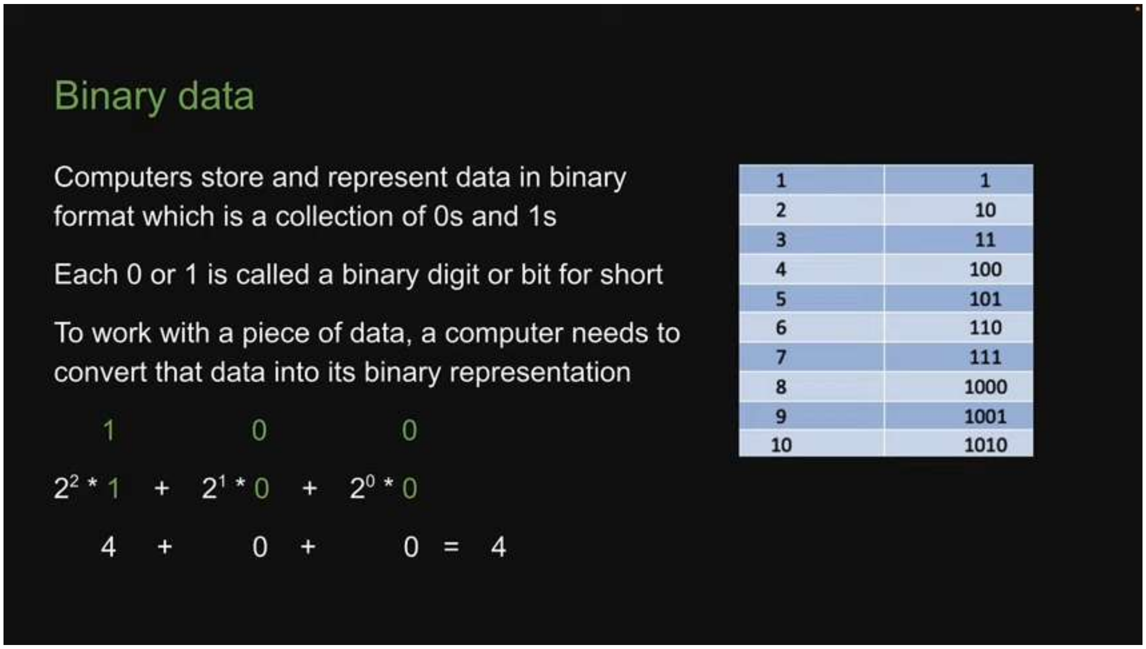
Detour

Character sets

Encoding

Streams and buffers

Asynchronous JavaScript



Binary data

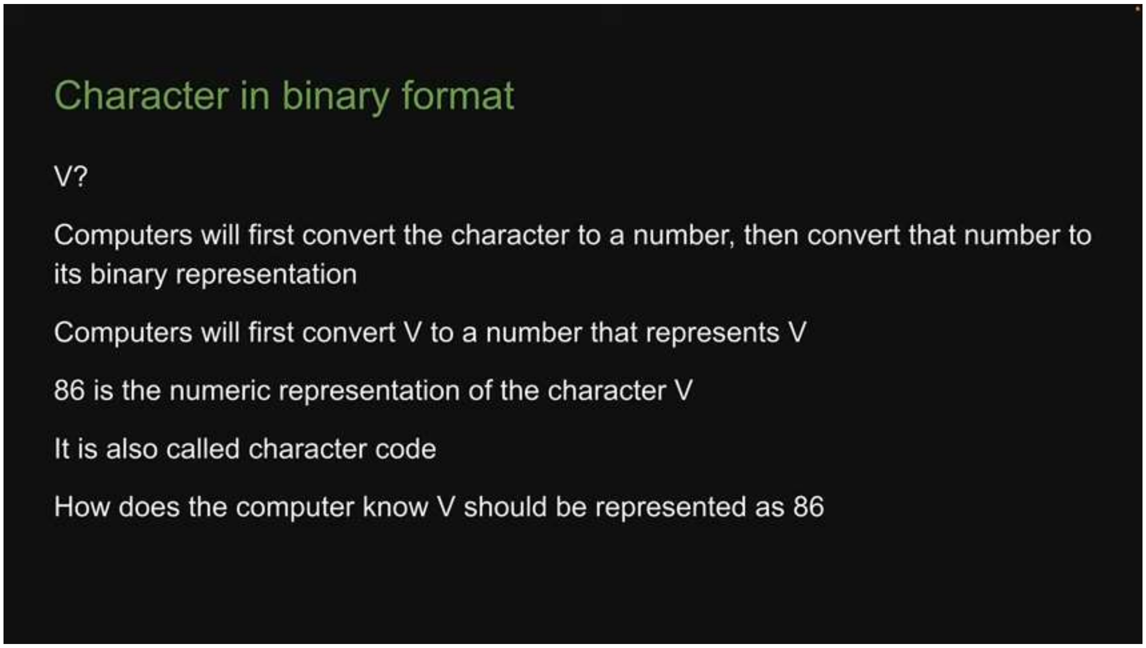
Computers store and represent data in binary format which is a collection of 0s and 1s

Each 0 or 1 is called a binary digit or bit for short

To work with a piece of data, a computer needs to convert that data into its binary representation

$$\begin{array}{ccccc} 1 & & 0 & & 0 \\ 2^2 * 1 & + & 2^1 * 0 & + & 2^0 * 0 \\ 4 & + & 0 & + & 0 = 4 \end{array}$$

1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010



Character in binary format

V?

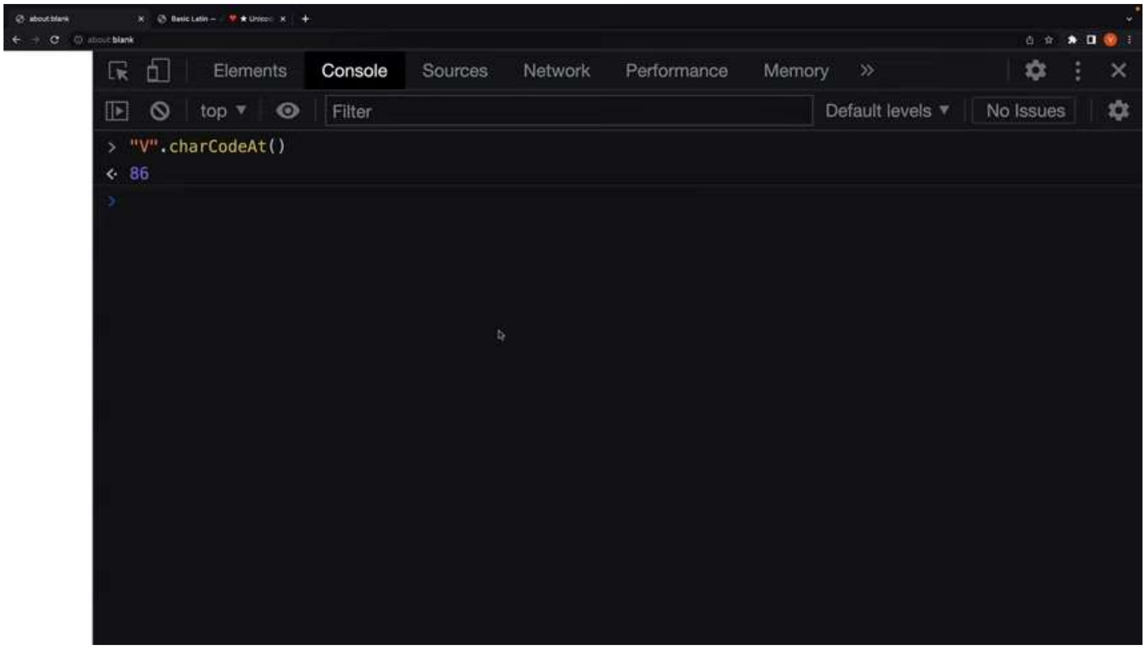
Computers will first convert the character to a number, then convert that number to its binary representation

Computers will first convert V to a number that represents V

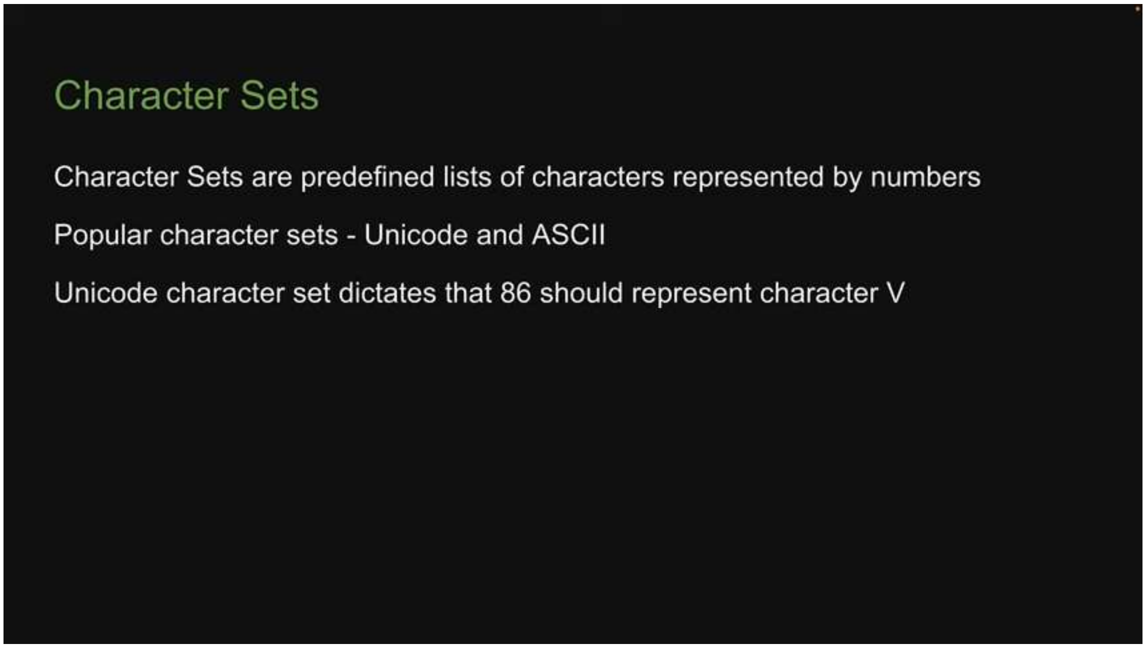
86 is the numeric representation of the character V

It is also called character code

How does the computer know V should be represented as 86



```
> "V".charCodeAt()
<· 86
>
```

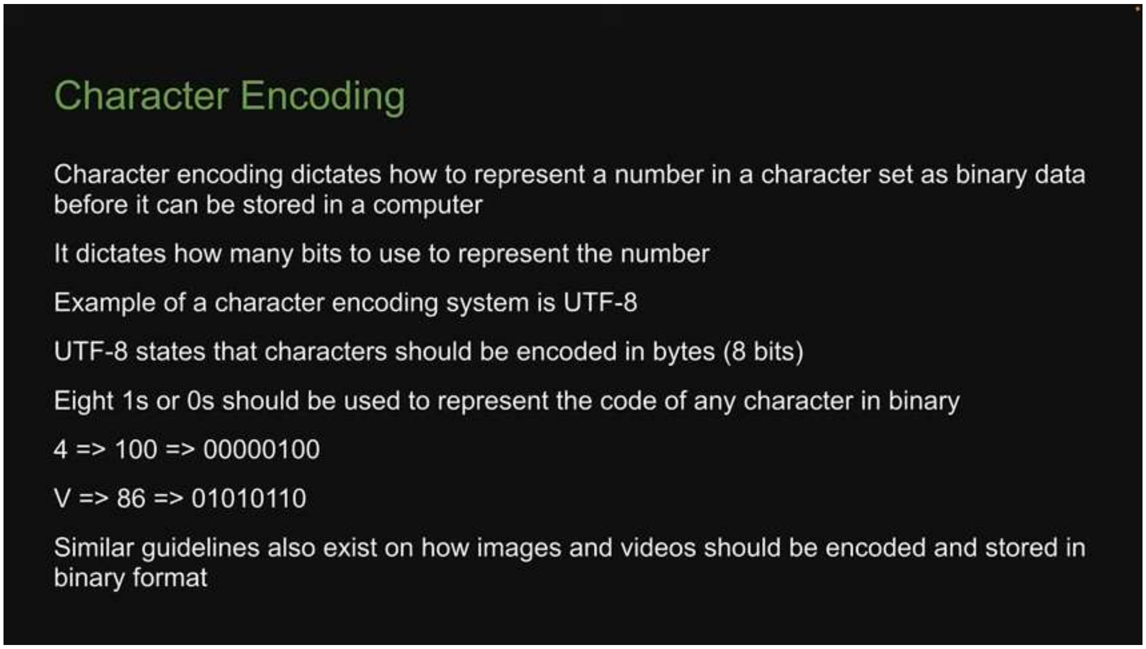


Character Sets

Character Sets are predefined lists of characters represented by numbers

Popular character sets - Unicode and ASCII

Unicode character set dictates that 86 should represent character V



Character Encoding

Character encoding dictates how to represent a number in a character set as binary data before it can be stored in a computer

It dictates how many bits to use to represent the number

Example of a character encoding system is UTF-8

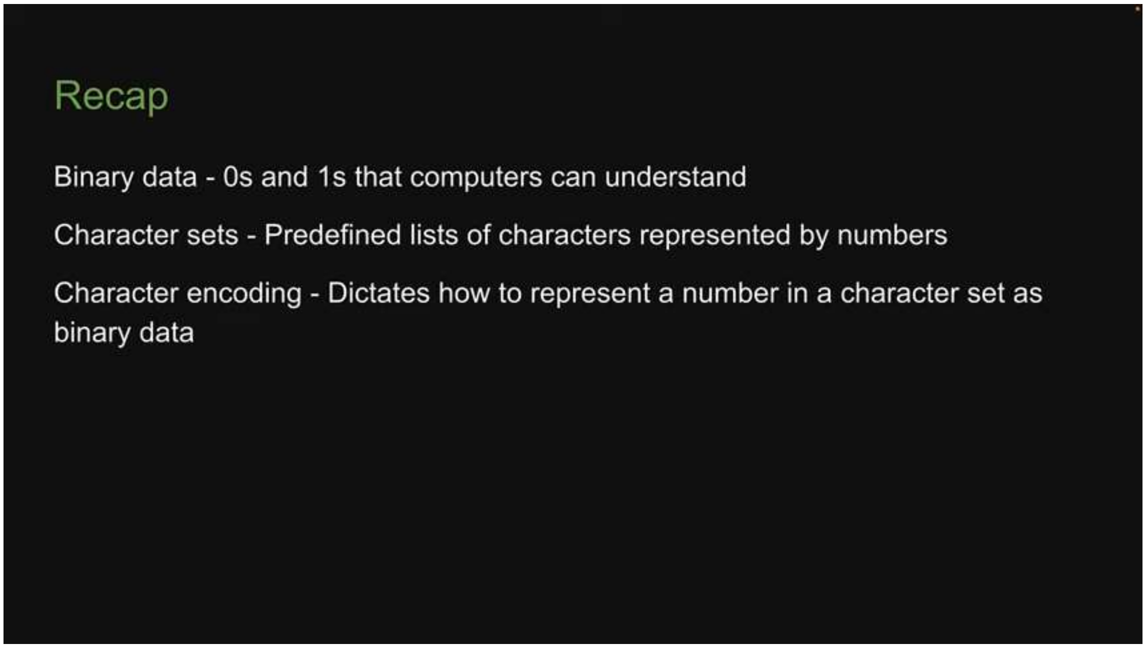
UTF-8 states that characters should be encoded in bytes (8 bits)

Eight 1s or 0s should be used to represent the code of any character in binary

4 => 100 => 00000100

V => 86 => 01010110

Similar guidelines also exist on how images and videos should be encoded and stored in binary format

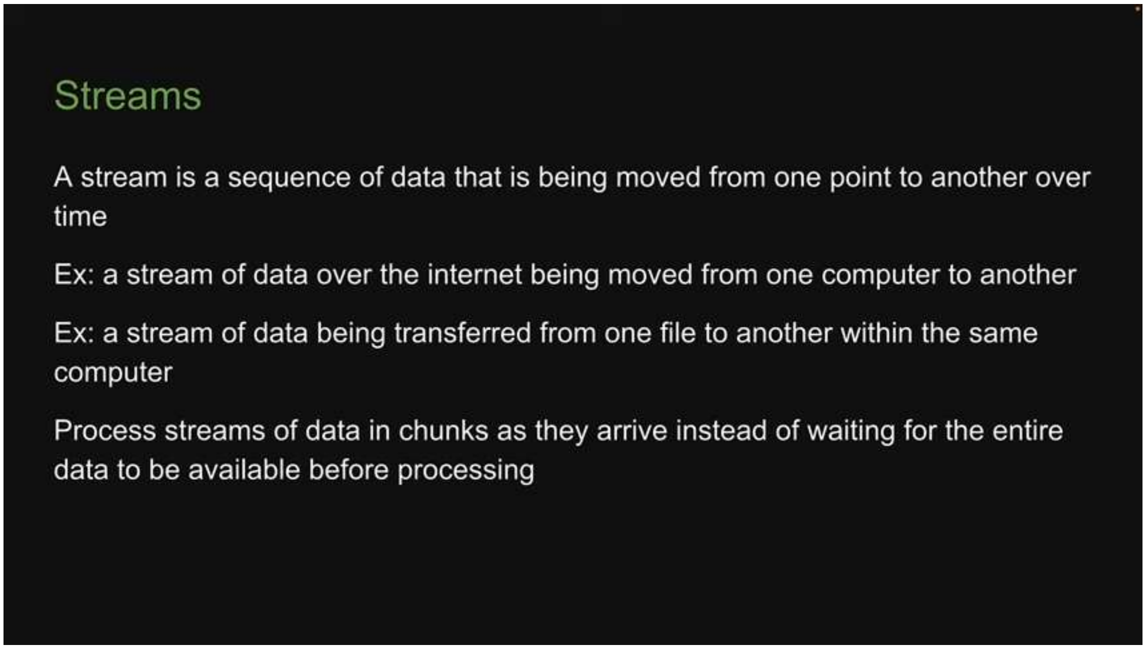


Recap

Binary data - 0s and 1s that computers can understand

Character sets - Predefined lists of characters represented by numbers

Character encoding - Dictates how to represent a number in a character set as binary data



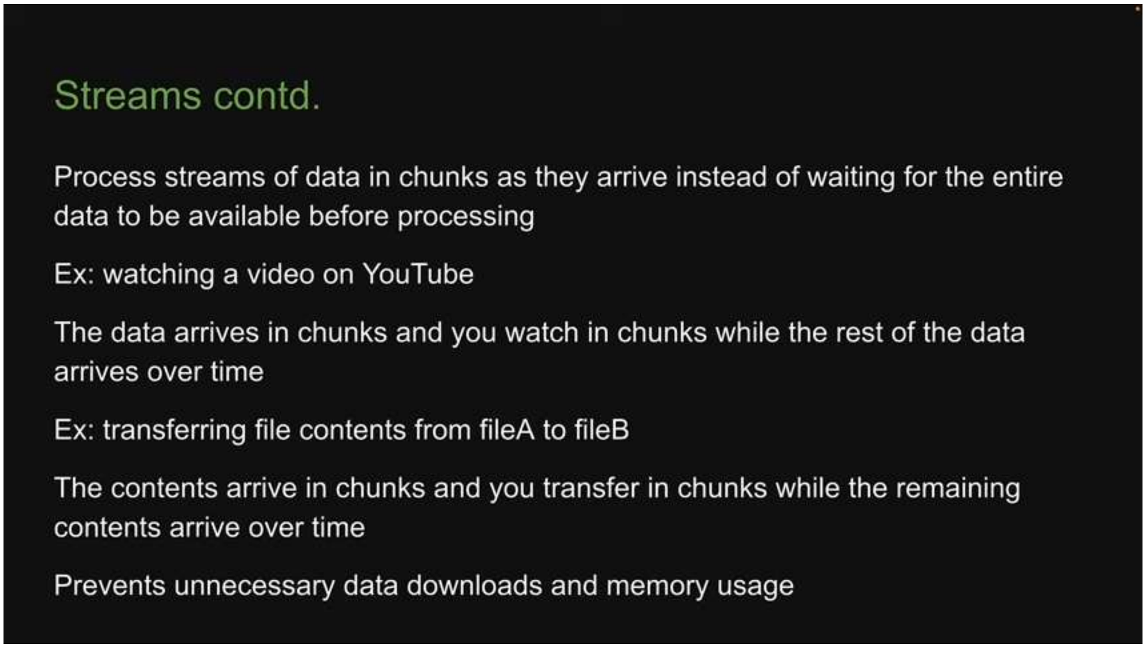
Streams

A stream is a sequence of data that is being moved from one point to another over time

Ex: a stream of data over the internet being moved from one computer to another

Ex: a stream of data being transferred from one file to another within the same computer

Process streams of data in chunks as they arrive instead of waiting for the entire data to be available before processing



Streams contd.

Process streams of data in chunks as they arrive instead of waiting for the entire data to be available before processing

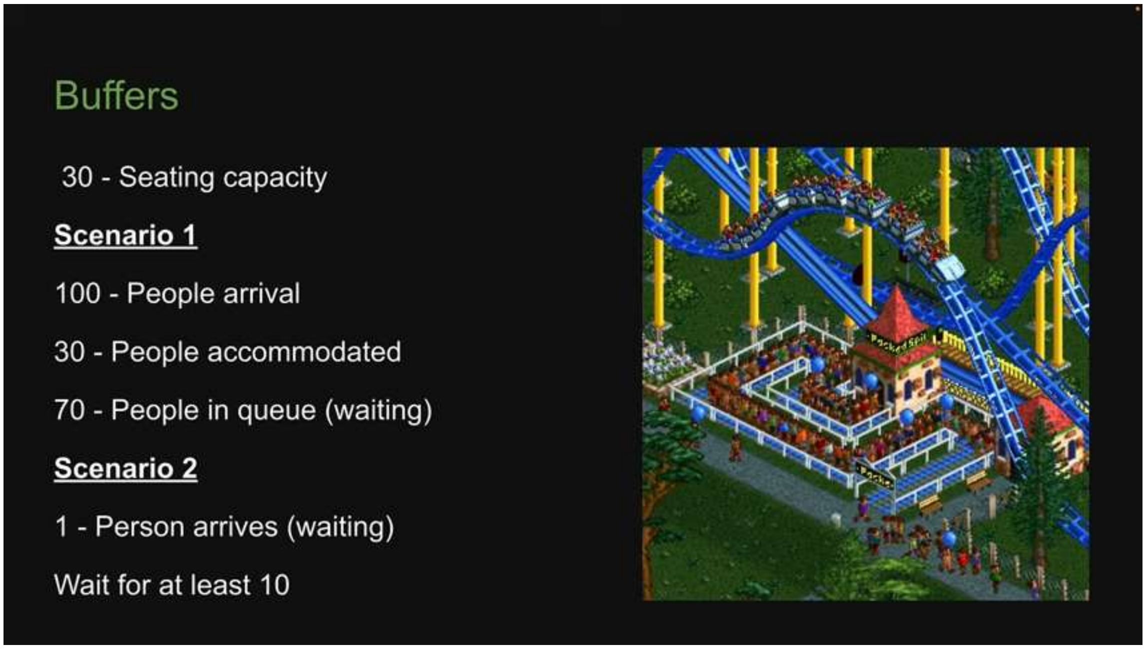
Ex: watching a video on YouTube

The data arrives in chunks and you watch in chunks while the rest of the data arrives over time

Ex: transferring file contents from fileA to fileB

The contents arrive in chunks and you transfer in chunks while the remaining contents arrive over time

Prevents unnecessary data downloads and memory usage



Buffers

30 - Seating capacity

Scenario 1

100 - People arrival

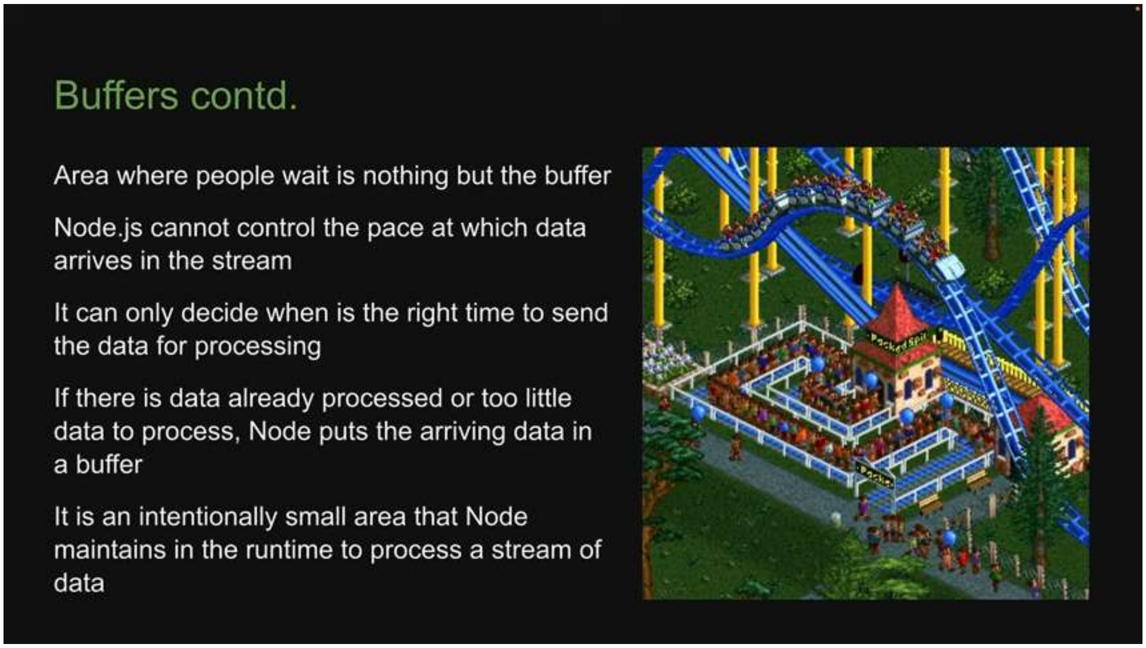
30 - People accommodated

70 - People in queue (waiting)

Scenario 2

1 - Person arrives (waiting)

Wait for at least 10



Buffers contd.

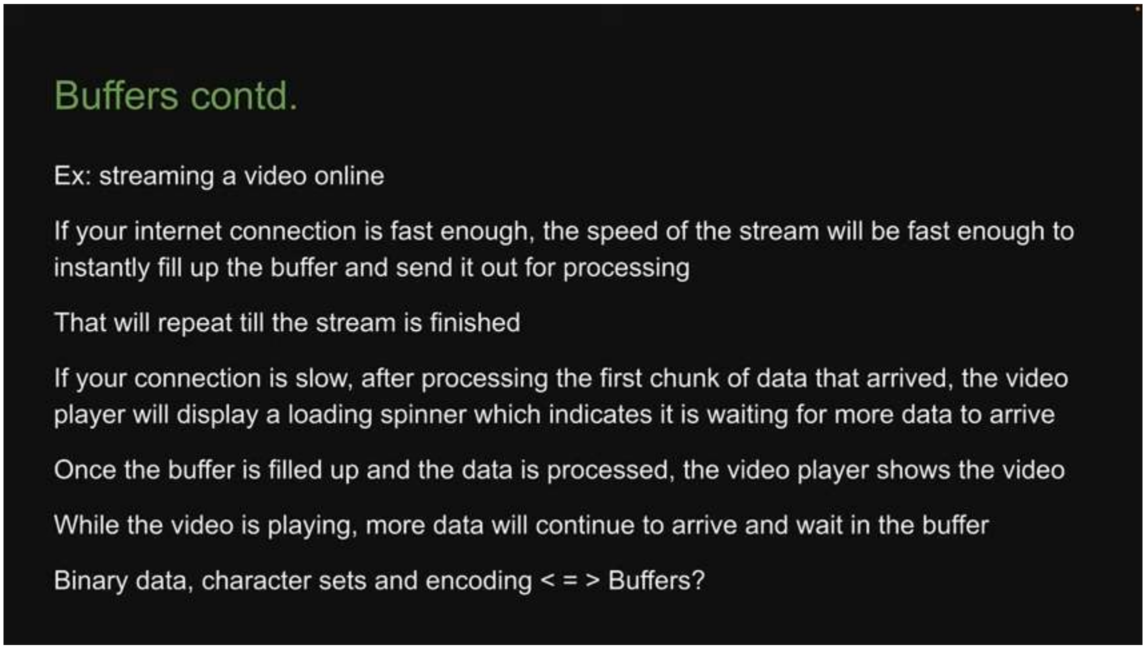
Area where people wait is nothing but the buffer

Node.js cannot control the pace at which data arrives in the stream

It can only decide when is the right time to send the data for processing

If there is data already processed or too little data to process, Node puts the arriving data in a buffer

It is an intentionally small area that Node maintains in the runtime to process a stream of data



Buffers contd.

Ex: streaming a video online

If your internet connection is fast enough, the speed of the stream will be fast enough to instantly fill up the buffer and send it out for processing

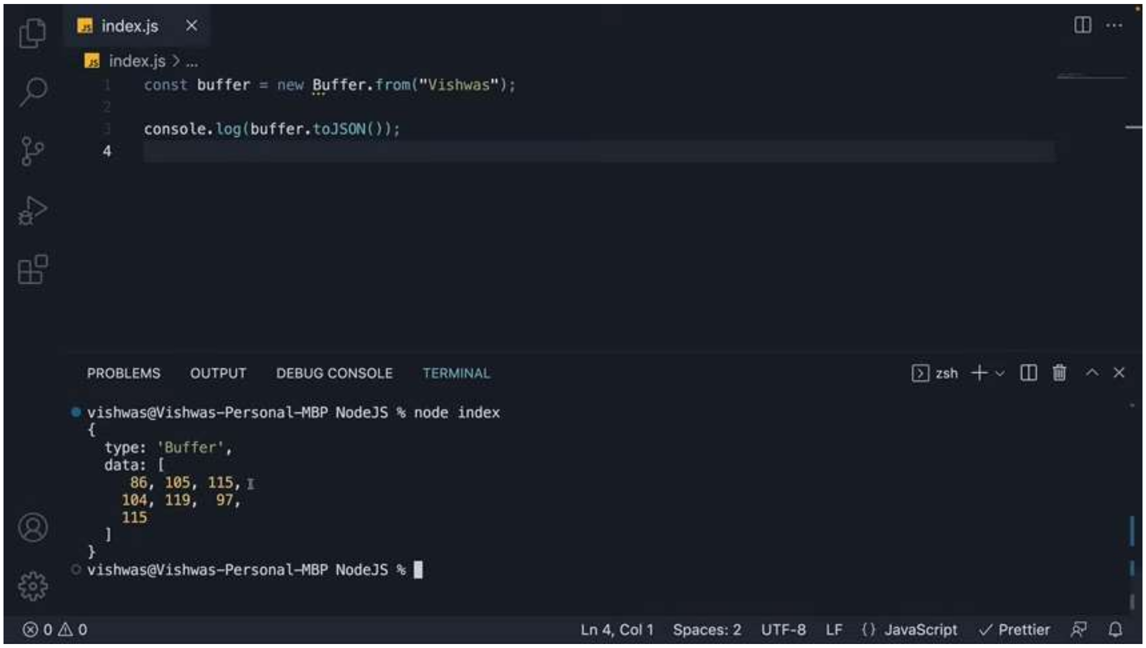
That will repeat till the stream is finished

If your connection is slow, after processing the first chunk of data that arrived, the video player will display a loading spinner which indicates it is waiting for more data to arrive

Once the buffer is filled up and the data is processed, the video player shows the video

While the video is playing, more data will continue to arrive and wait in the buffer

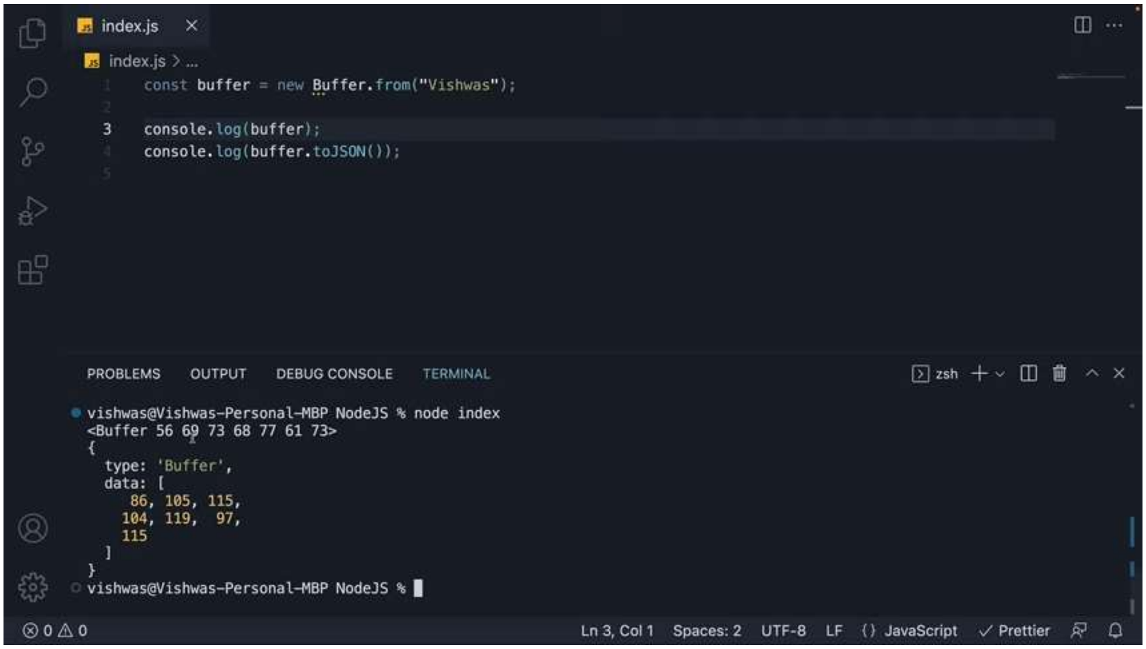
Binary data, character sets and encoding < = > Buffers?



```javascript
const buffer = new Buffer.from("Vishwas");

console.log(buffer.toJSON());
```

```
vishwas@Vishwas-Personal-MBP NodeJS % node index
{
  type: 'Buffer',
  data: [
     86, 105, 115,
    104, 119,  97,
    115
  ]
}
vishwas@Vishwas-Personal-MBP NodeJS %
```



```javascript
const buffer = new Buffer.from("Vishwas");

console.log(buffer);
console.log(buffer.toJSON());
```

```
vishwas@Vishwas-Personal-MBP NodeJS % node index
<Buffer 56 69 73 68 77 61 73>
{
  type: 'Buffer',
  data: [
     86, 105, 115,
    104, 119,  97,
    115
  ]
}
vishwas@Vishwas-Personal-MBP NodeJS %
```