Session 5.0

Node Fundamentals - II

Topics

- **Node Fundamentals - II**
  - Built-In Modules
  - Buffers and Streams
  - Files and Streams

Buffers and Streams

Buffer

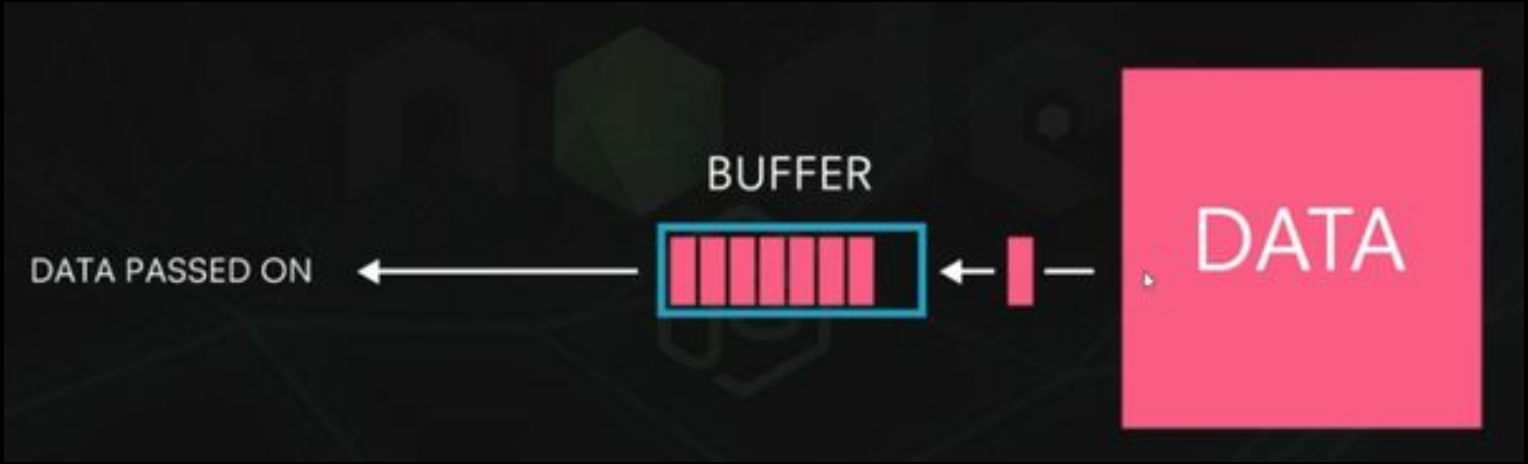
- A temporary holding spot for data being moved from one place to another.
- The buffer is filled with data, then passed along
- Transfer small chunks of data at a time without waiting for whole data to download. ie. Youtube video

Buffer

- A buffer is a raw set of data from memory with no defined type. That means it can be anything:

    - Text file
    - Video
    - Image
    - An array

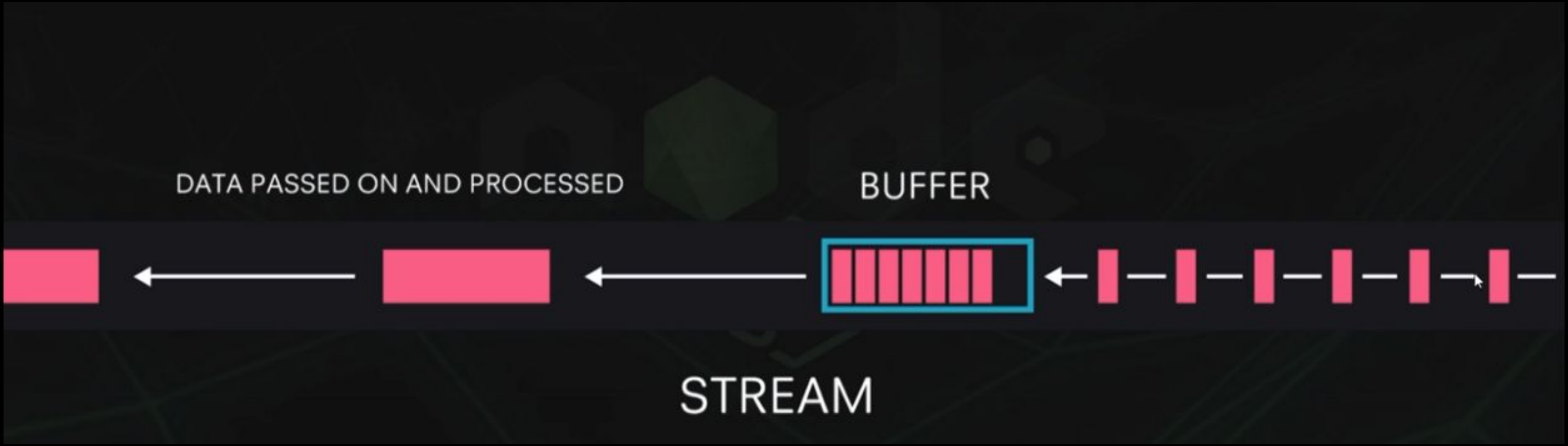
Buffer



Streams

- Stream is just a process of flow of data from the data source to the buffer and from the buffer to the client.
- All these data ‘chunks’ flow in a stream to transfer data
- Used to increase performance

Streams



Built-In Node Modules

The “util” module

https://nodejs.org/docs/latest/api/util.html

Provide information utilities about the current running system. It can be accessed by

const util = require(‘util’);

- debuglog()
- deprecate()
- format()
- inherits()
- inspect()

** inherits() will join the prototype from one object to another

The “os” module

https://nodejs.org/api/os.html

Provide information utilities about the current running system. It can be accessed by

const os = require(‘os’);

- os.tmpDir()
- os.hostname()
- os.type()
- os.platform()
- os.arch()
- os.release()

- os.uptime()
- os.loadavg()
- os.totalmem()
- os.freemem()
- os.cpus()
- os.networkInterfaces()
- as.EOL

Files and Streams

Streams

- Streams are instance of and extensions to the EventEmitter

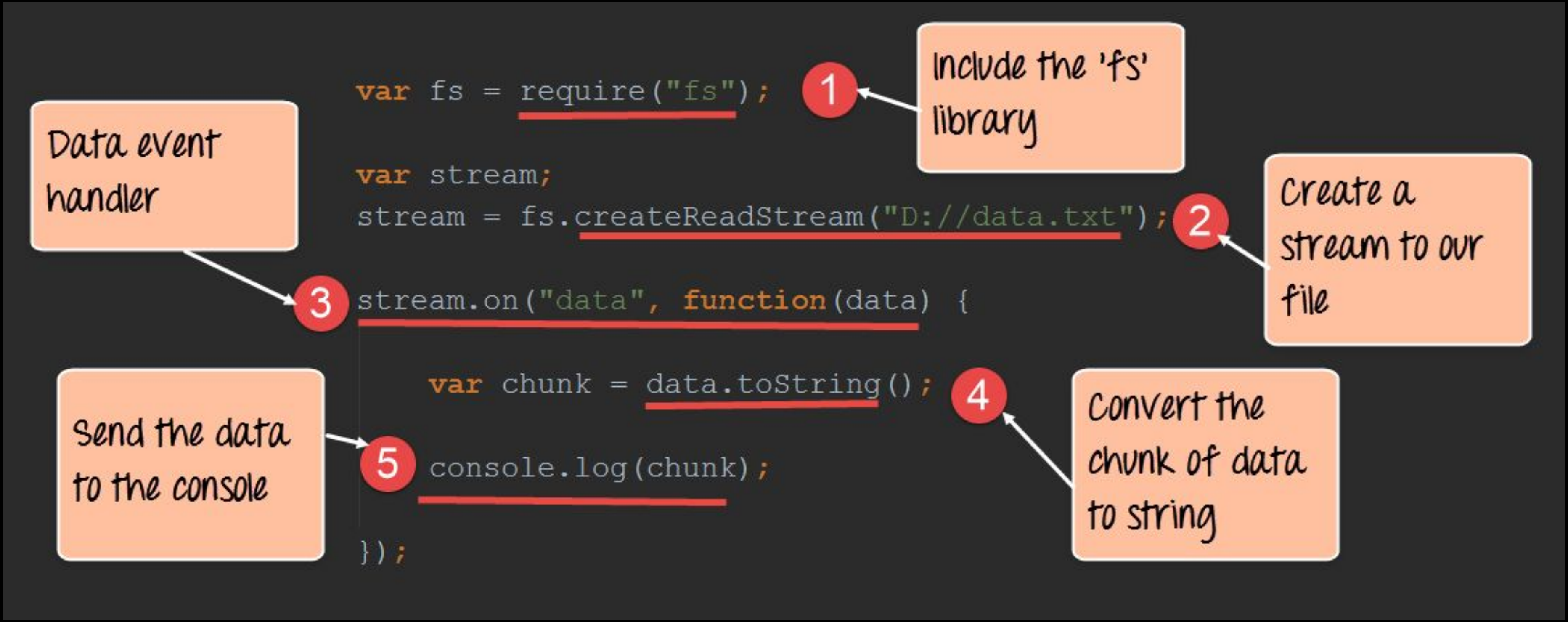
- Used for managing data flow, including
  - Network traffic (http requests & responses, tcp sockets)
  - File I/O
  - stdin/stdout/stderr

- Streams can be either readable, writable or both!

Streams

- Writable streams - can write data to a stream
- Readable streams - can read data from a stream
- Duplex - can read and write to a stream

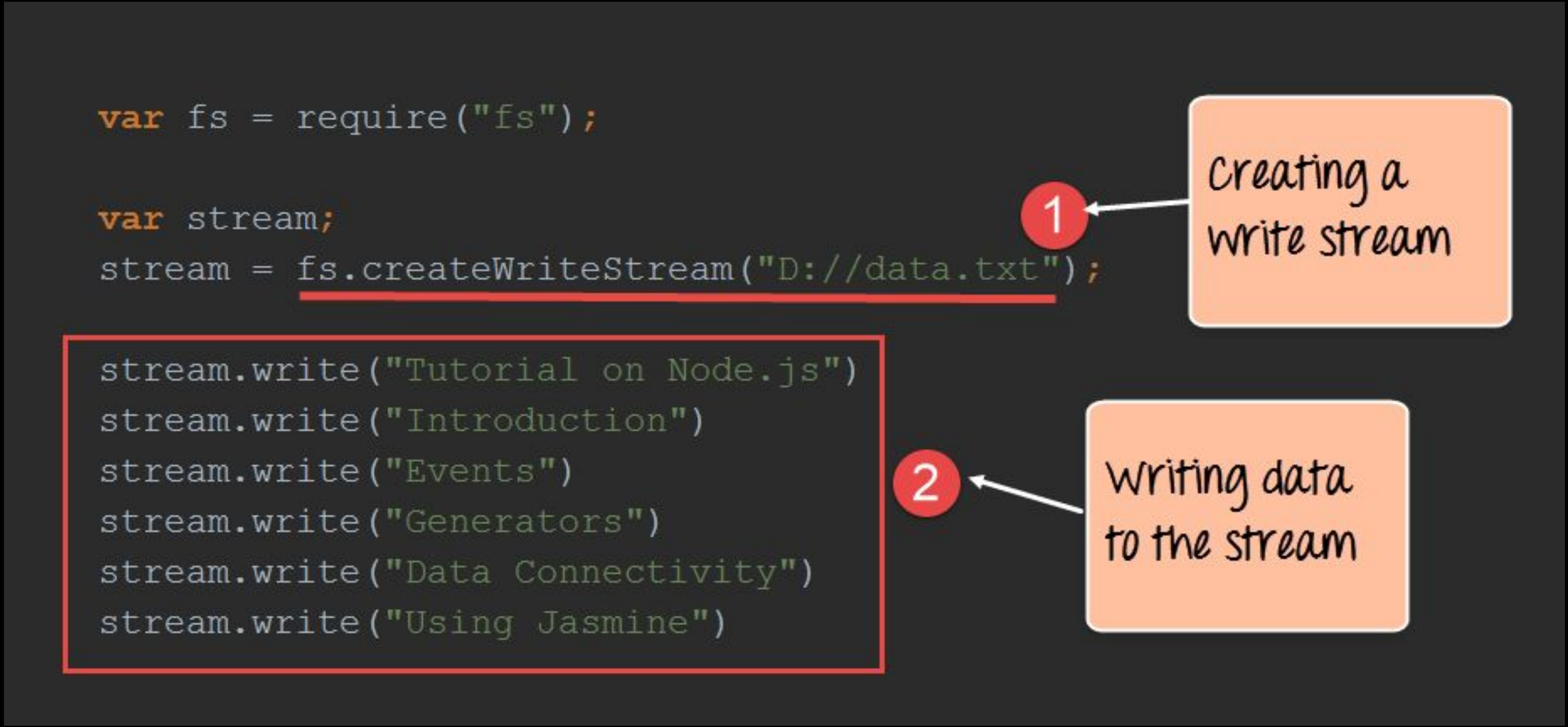
Creating Readable Stream



```
var fs = require("fs");

var stream;
stream = fs.createReadStream("D://data.txt");

stream.on("data", function(data) {

    var chunk = data.toString();

    console.log(chunk);

});
```

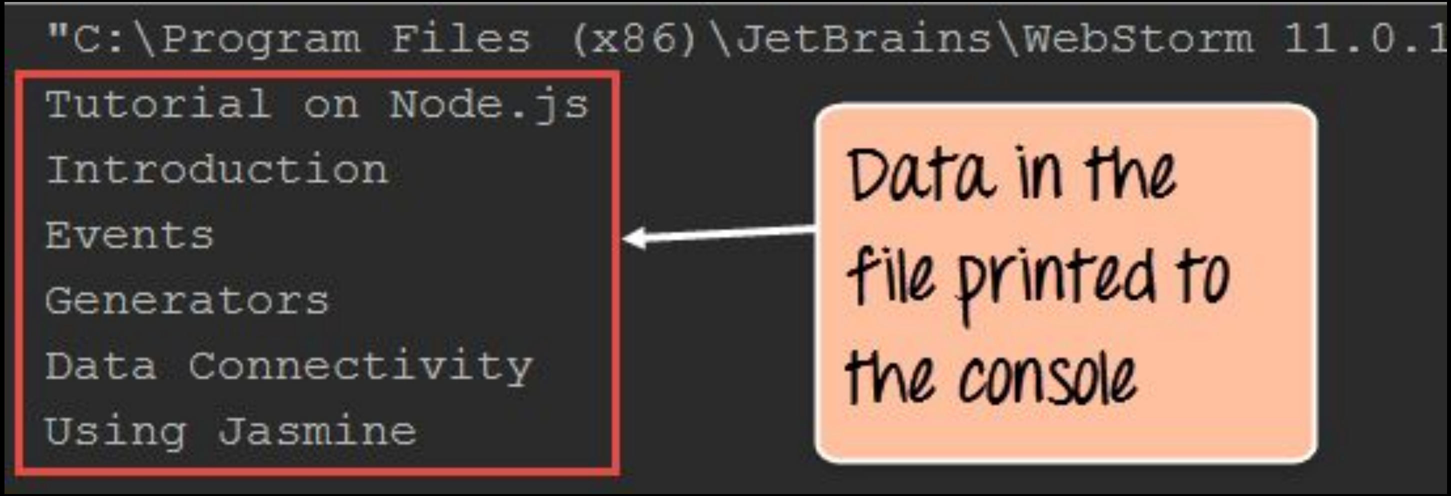
1
Include the 'fs' library
2
Create a stream to our file
3
Data event handler
4
Convert the chunk of data to string
5
Send the data to the console

Creating Writable Stream



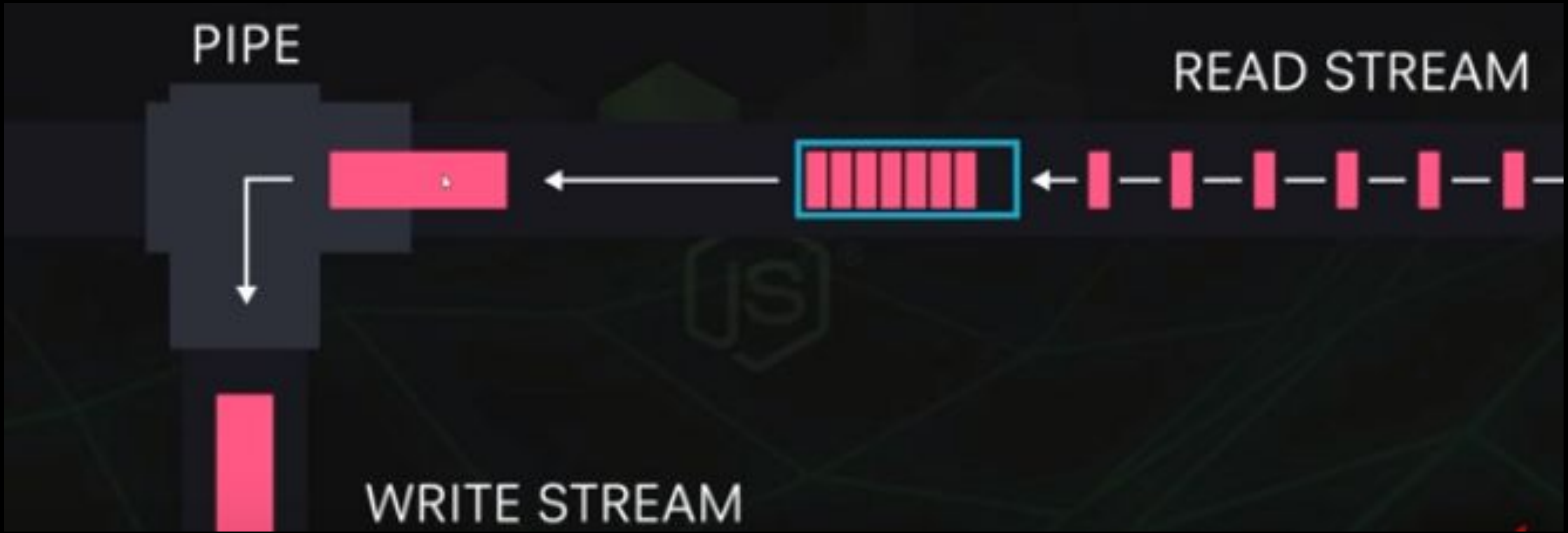
```
var fs = require("fs");

var stream;
stream = fs.createWriteStream("D://data.txt");

stream.write("Tutorial on Node.js")
stream.write("Introduction")
stream.write("Events")
stream.write("Generators")
stream.write("Data Connectivity")
stream.write("Using Jasmine")
```

Output from Writable Stream

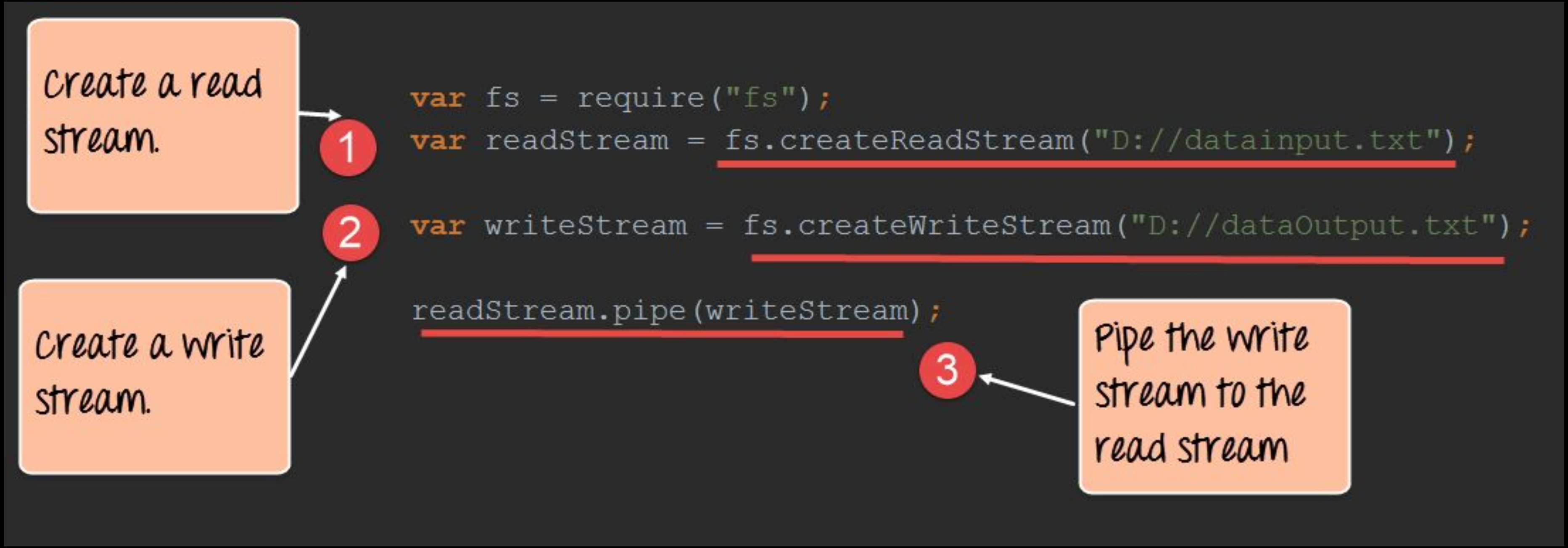


```
"C:\Program Files (x86)\JetBrains\WebStorm 11.0.1
Tutorial on Node.js
Introduction
Events
Generators
Data Connectivity
Using Jasmine
```

Pipe

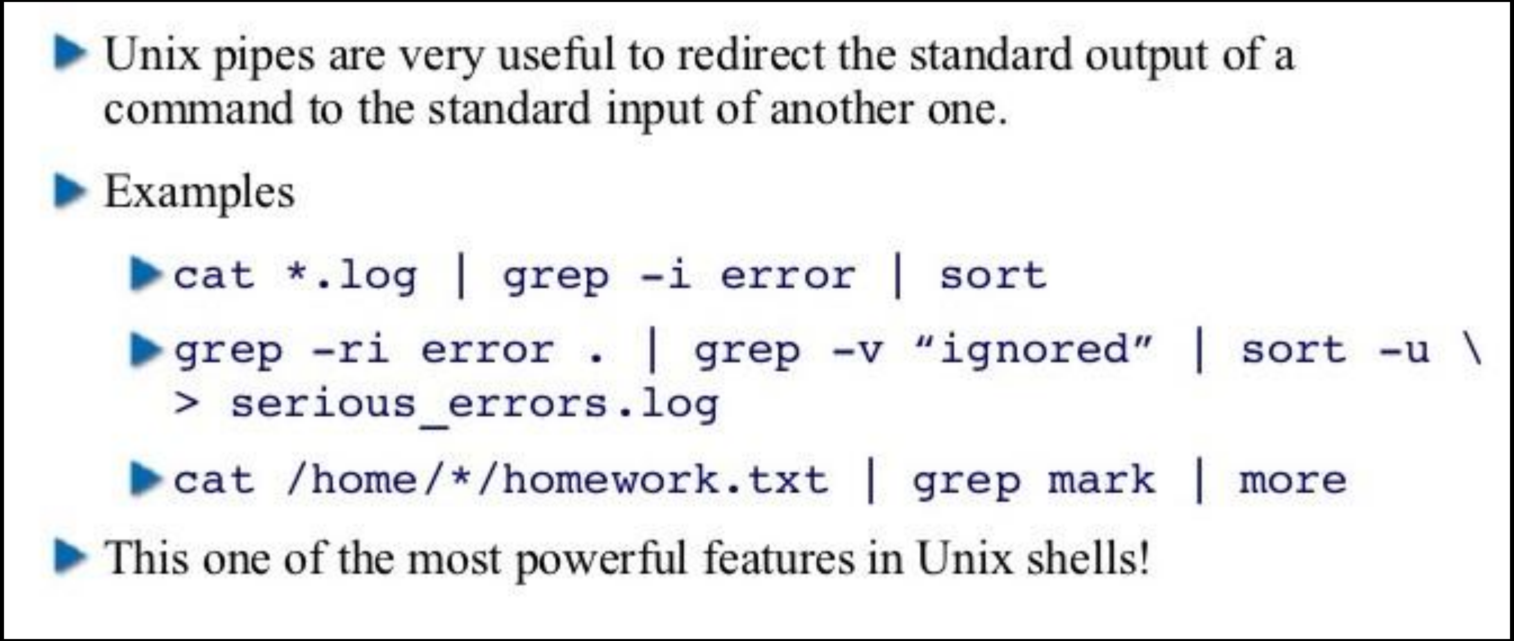


Piping between two streams



```
var fs = require("fs");
var readStream = fs.createReadStream("D://datainput.txt");

var writeStream = fs.createWriteStream("D://dataOutput.txt");

readStream.pipe(writeStream);
```

Nodejs through Libuv is using the old UNIX commands



- Unix pipes are very useful to redirect the standard output of a command to the standard input of another one.
- Examples
  - `cat *.log | grep -i error | sort`

  -
    ```
    grep -ri error . | grep -v “ignored” | sort -u \
    > serious_errors.log
    ```
  - `cat /home/*/homework.txt | grep mark | more`
- This one of the most powerful features in Unix shells!

Video - Intro to Streams