**DAY 1**

**How Node JS differs from Vanilla JS?**

* Node runs on server --- not browser
* The console in node is terminal window – not browser console
* Has global object instead of windows object
* By default Node has CommonJS modules instead of ES6 modules.
* Missing some APIs like fetch API

--dirname property

--filename property

**OS module:**

* Os.type()
* Os.version()
* Os.homedir()

**Path module:**

* Path.dirname(\_\_filename)
* Path.basename(\_\_filename)
* Path.extname(\_\_filename)
* Path.parse(\_\_filename)
* Path.join() method- recommended to use for file paths

fs.readFile(path.join(\_\_dirname, "files", "starter.txt"), (err, data) => {

  console.log(data.toString());

  if (err) throw err;

});

**fs module:**

* Fs.readFile(),
* Fs.writeFile(),
* Fs.appendFile(),

**DAY 2**

**Promises in Fs module:**

* Import promises and use the async functionality of all operations

const fsPromises = require("fs").promises;

* **Delete a file:**
* await fsPromises.unlink(path.join(\_\_dirname, "files", "starter.txt"));
* **Rename a file:**
* await fsPromises.rename(
* path.join(\_\_dirname, "files", "promiseWrite.txt"),
* path.join(\_\_dirname, "files", "promiseComplete.txt")
* );
* Use **fs.existsSync()** method to check weather a directory or a file exists or not
* Make new directory using **fs.mkdir()**  method.
* Delete a directory using **fs.rmdir()**  method.

**Streams- a better way to read and write files:**

fs.createReadStream()

fs.createWriteStream()

**Node Modules**

* NPM
* Installing modules / removing them
* Adding dev dependencies, scripts in package.json file

**DAY 3**

* Event Emitters