

**Introduction to Node.js**

**File System module**

**“There is not now, nor has there ever been, nor will there ever be, any programming language in which it is the least bit difficult to write bad code.”**

**Flon’s Law**

# Pokemon Assignment

## First steps

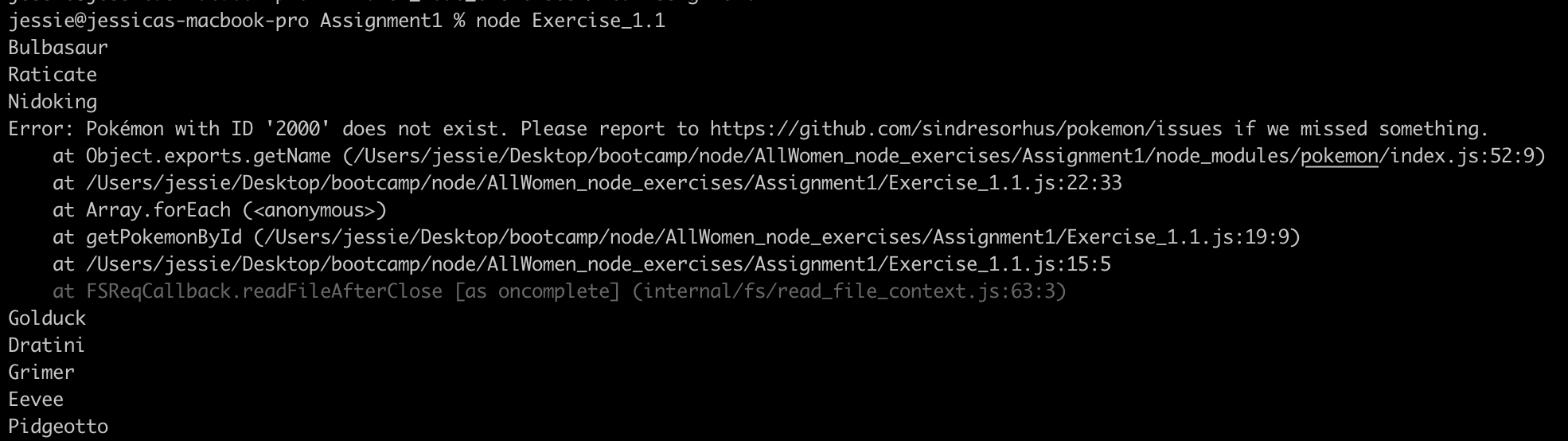
This assignment consists of three exercises focused on the practice of the File System native module of Node.js and the node package manager (npm) we have seen in class. Before moving onto the practice, please follow the instructions below:

1. *First of all,* create a folder called *Assignment 2* on your Git Repository to commit all the exercises.
2. Create a folder called *Assignment 2* (try to do it by command line)
3. Each exercise should be in its own file, so let’s start creating a file called *exercise\_1.1.js*
4. Initialize npm with the command **npm init**
5. Install the package by typing in your terminal **npm install pokemon**
6. Take a look at [Pokemon API](https://www.npmjs.com/package/pokemon) (methods, params …)
7. Open your IDE executing the command **code .** If this doesn’t work for you, take a look at [Visual Studio Launching from Command Line](https://code.visualstudio.com/docs/editor/command-line) or open the IDE’s application.
8. You should commit each exercise to your repository (a new one or some repo that you already have, this decision is up to you).

## Exercise 1.1

Your mission is to print in the terminal the name of the Pokemons whose ID’s are defined in the file *ids\_pokemon.txt* (\*\* you first need to create this file manually and add some ids to this file for testing \*\*) taking into account the following restrictions:

* Read the file **synchronously**. Hint: Take a look at fs.readFileSync(path[, options]) from the File System of node.js native module.
* What is the [***\_\_dirname*** global variable](https://www.digitalocean.com/community/tutorials/nodejs-how-to-use__dirname)? console log it to see it in action
* Error handling: Take into account the possible errors that could appear during the execution. For the purpose of this exercise, print them as well then will know what’s happening. Tip: catch the errors enclosing your code in a try-catch statement!
* The expected output is as follows:



## Exercise 1.2

In the last exercise, you used the function fs.readFileSync to read the file *ids\_pokemon.txt*. For this exercise, we propose you to do exactly the same but reading the file **asynchronously.** Hint: Take a look at fs.readFile(path[, options], callback).

## Exercise 1.3

This time instead of printing the names on the terminal you will write them down into a file called *names\_pokemon.txt* taking into account the following restrictions:

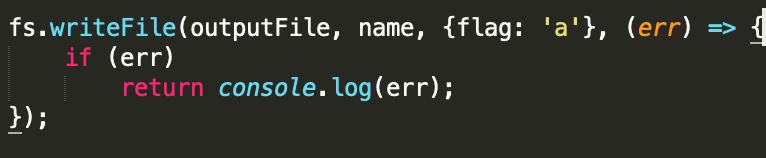
* Write **all** the names on the file. Hint: take a look at fs.appendFile(path, data[, options], callback)
* The expected output (names could change):



# The Curious Corner

In this section you will find some extra reading regarding some of the problems, doubts you may have encountered during the assessment.

* What is the main difference between fs.readFileSync and fs.readFile?
* [Blocking vs Non-Blocking methods](https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/)
* Many of the fs methods return or require a *buffer*. What is the Buffer object? How do I get a string from a buffer? (hint; it is raw binary data, so use *toString()*, or specify encoding UTF-8 to get something readable)
  + [Node.js Buffers](https://nodejs.dev/learn/nodejs-buffers)
* How are errors handled in Node.js?
* [Errors official documentation of Node.js](https://nodejs.org/api/errors.html)
* [Error handling Guide with defined uses cases](https://nodejs.dev/learn/error-handling-in-nodejs)
* How to embed expressions (as var1) into string literals?
* [Template literals](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Template_literals)
* Is there any other way to keep writing into a file as we did in the exercise 1.3? Yes there is, we could use fs.writeFile(file, data[, options], callback) using the flag ‘a’ (append) as it shown in the following snippet:



For more info take a look to the following link: [Write or append a file](https://egghead.io/lessons/node-js-write-or-append-to-a-file-in-node-js-with-fs-writefile-and-fs-writefilesync)

* Regarding exercise 1.2, did you notice that depending on where you place your try/catch statement you will obtain a different output?
* *How to*…
* Read files in node.js: [Tutorial 1](https://nodejs.org/en/knowledge/file-system/how-to-read-files-in-nodejs/), [Tutorial 2](https://nodejs.dev/learn/reading-files-with-nodejs)
* Write files in node.js: [Tutorial 1](https://nodejs.org/en/knowledge/file-system/how-to-write-files-in-nodejs/), [Tutorial 2](https://nodejs.dev/learn/writing-files-with-nodejs)