**Full Stack Development lll– Lab 1**

* NPM – Node Package Manager

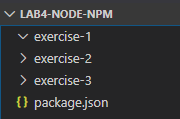
**Node Reference**

An optional reference for Node JS

**https://www.w3schools.com/nodejs**

**Developer Note:**When working on your exercises, please create seperate folder for your work. This way you won’t putting all your code in the same file, which can pollute the global namespace. In short, it will prevent you from overwriting your own work and causing your code to compile incorrectly.

Organize your folder structure in this way.



**Task 1: Installing Node.js**

Follow one of the following tutorials to install Node.js on your local environment.   
(select one based on your operating system ie. windows)

<https://nodesource.com/blog/installing-nodejs-tutorial-windows>

<https://nodesource.com/blog/installing-nodejs-tutorial-mac-os-x>

<https://nodesource.com/blog/installing-node-js-tutorial-ubuntu>

<https://nodesource.com/blog/installing-node-js-tutorial-debian-linux>

**Installation tutorial for mac os video**

<https://youtu.be/TmT_CGFnUuM>

**Installation tutorial for windows video**

<https://youtu.be/AuCuHvgOeBY>

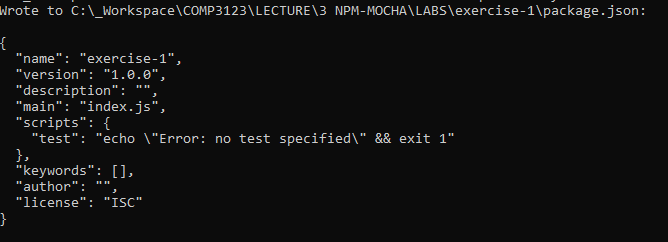
**Exercise 1**

1. Open a command prompt create a directory for **exercise-1**
2. Open Visual Studio Code and open the folder **exercise-1**
3. In command prompt go to the directory and run the following **npm init** command to create a **package.json** file

<https://docs.npmjs.com/cli/init>



The console will return the following text contained in the newly created **package.json** file

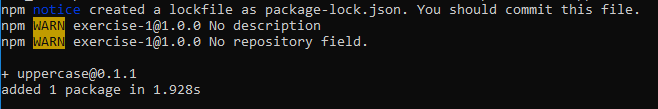


1. Next execute the following command to install 3rd party module from npm named **uppercase**.

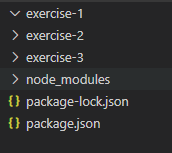
<https://www.npmjs.com/package/upper-case>



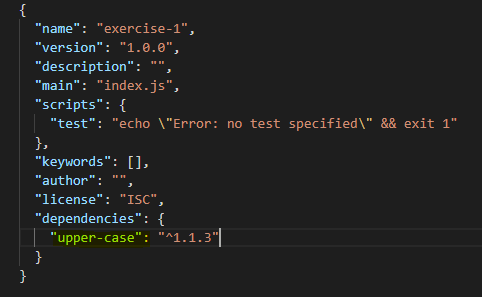
The console will return the following output



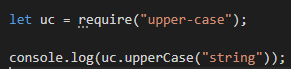
The folder structure will look as follows. The **package-lock.json** can be deleted.



The module reference to uppercase will now be listed as a **dependency** in the **package.json** file



1. Create an **ex1.js** file and write the following code.

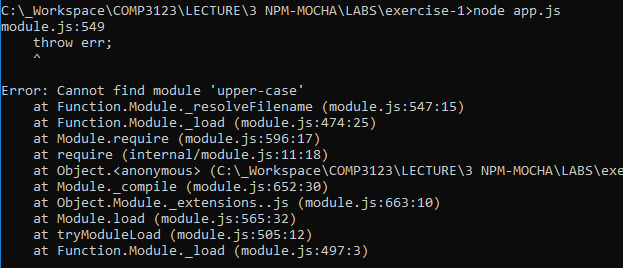


1. Run on the command line with **node ex1.js** and you will get the following output.



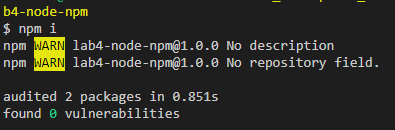
1. Browser your froot older structure and delete the **node\_modules** folder

1. Repeat step 6 to run the **ex1.js** again and you will get the following error.



1. \* This is what the folder structure will look like when we deploy or submit our code. The **node\_module** will not be part of deployment. The **package.json** contain all the dependencies needed for your application.

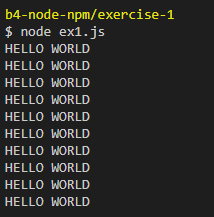
Use the **npm -install** in the root folder command and the **upper-case** module will be restored.



1. The **node\_module** folder will now be restored and when you run execute the **app.js** it will compile again.

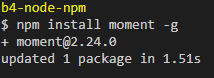


Task: Create a function named greeter in app.js that will use upper case and output hello world 10 times and log it to the console.

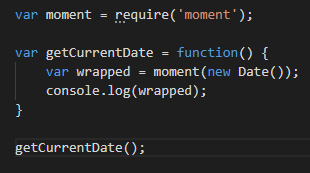


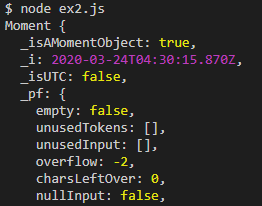
**Exercise 2: Using require and moment.js**

1. Download moment.js using npm



1. In the **exercise 2** folder, create a new file named **ex2.js** using require to make **moment.js** available in your code and run using node. The output will be a large moment date object





1. Task 1: Convert the javascript to **ES6** syntax
2. Task 2: Using the **moment.js** documentation find a way to parse the date and time into separate strings and output to console. <https://momentjs.com/>

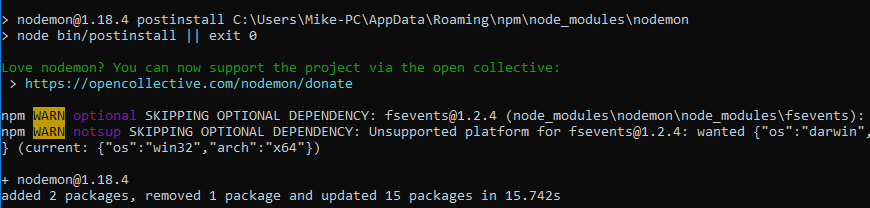


**Exercise 3: Nodemon + NPM Registry modules**

* 1. Open a command prompt create a directory for **exercise-3**
  2. Open Visual Studio Code and open the folder **exercise-3**
  3. Install **nodemon** (node monitor) from NPM using **install** using **–global** flag   
     <https://www.npmjs.com/package/nodemon>

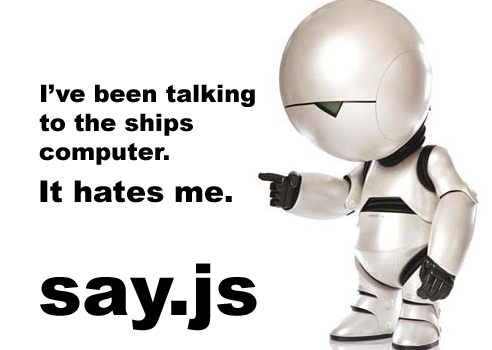


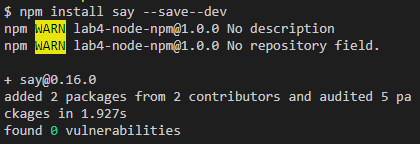
This will **install nodemon** as a **global package** of Node on your computer to be used everywhere in all projects you create.



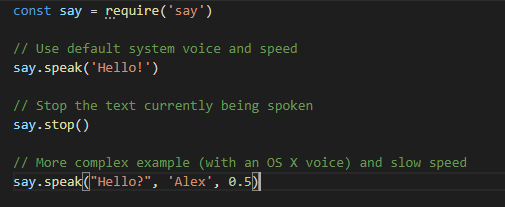
* 1. Install module named **say.js** from NPM Registry using install and **–save** flag

<https://www.npmjs.com/package/say>

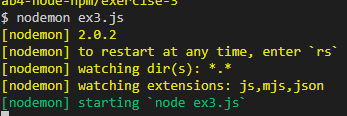




* 1. Create an **ex3.js** file (file name does not matter as long as it is in the exercise 3 folder) and write the following code



* 1. Use the newly installed **nodemon** to run the **app.js**. The text should be played through on your pc system voice.



* 1. Create a new callback function that will use the **say.speak** command   
     ie. A function named sorryDave speaks the text “I’m sorry, Dave”

**Challenge:** Use the **setTimeOut** function to trigger a callback after a delayed amount of time ie. 5 seconds, so that the text is spoken after the first “Hello, Alex”

<https://javascript.info/settimeout-setinterval>