Installing and Running Node Modules

1. Intro to Node.js

Node.js is an asynchronous event driven JavaScript runtime environment.
It's NOT a new programming language.
It's just the runtime environment that runs JavaScript programs.
It allows running JavaScript on a server.

In this tutorial, we'll install Node and a Node module using **npm.**

We'll learn:

a) Installing Node packages
b) Creating the package.json file
c) Running a node app from the terminal window

2. Installing Node and NPM

1. If you're using your local computer, go to:
   https://nodejs.org/en/download/
   And download the latest version for your operating system.

2. If you're using AWS Cloud 9, node and npm are already installed. Open a Terminal window (Window > New Terminal) and type the following two commands to see their corresponding versions:

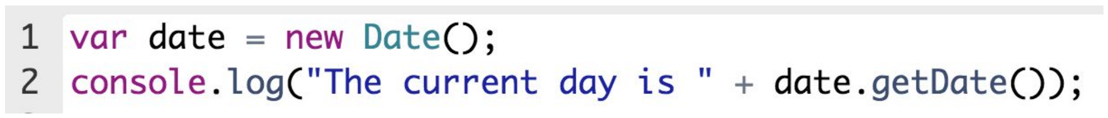
```
node --version
npm --version
```

3. Running an app in Node

1. Each app should be included within its own folder. Create a folder called “**firstNodeApp**” within your “**practice**” folder (this is not a lab)

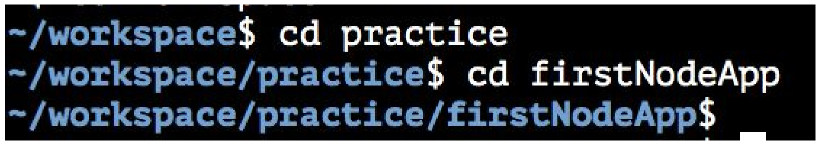
2. Within the “**firstNodeApp**” folder, create a file called “**app.js**” (it can be any name but we’ll use this one by convention)

3. Type any JavaScript code, just output something to the console, such as:



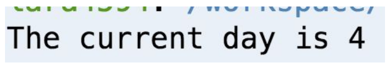
```
var date = new Date();
console.log("The current day is " + date.getDate());
```

4. Open a new Terminal window and change directories to go to the **firstNodeApp** folder.



```
~/workspace$ cd practice
~/workspace/practice$ cd firstNodeApp
~/workspace/practice/firstNodeApp$
```

5. Type:
   `node app.js`
   This should run the program and you should see something like the following output. You can run now any JavaScript program using node!



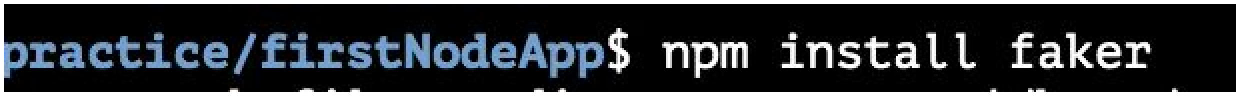
```
The current day is 4
```

4. Installing a Node Package

There are hundreds of thousands of packages (or libraries) that you can install and run in node. To install modules, you need to use the "npm" command. "npm" stands for Node Package Manager. npm is the world's largest software registry. You can search for npm packages at: https://www.npmjs.com/

We'll install a package called "faker" that generates random fake data. It's located here: https://www.npmjs.com/package/faker

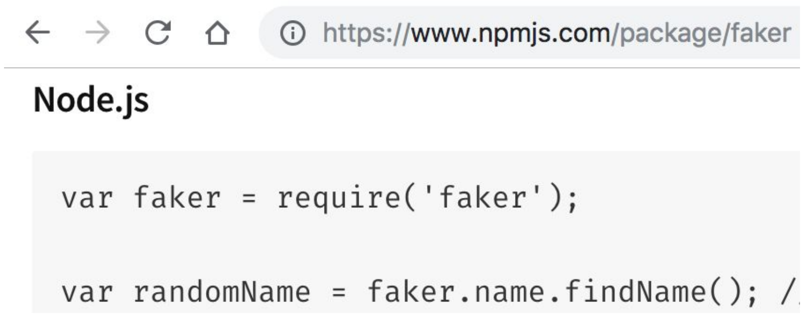
1. Within the "**firstNodeApp**" folder, type:



```
practice/firstNodeApp$ npm install faker
```

   Note: Most likely, you'll get a warning about a missing "package.json" file. Ignore it for now. We'll create it later. Without this file, Cloud 9 install the packages into a temporary folder.

2. The "faker" package documentation provides examples of the methods that you can use and the way to import the library. Type the following two lines of code (provided by the documentation) in the **app.js** file (delete the previous two lines we had there)



https://www.npmjs.com/package/faker

Node.js

```
var faker = require('faker');

var randomName = faker.name.findName(); /
```

3. The new first line of code:

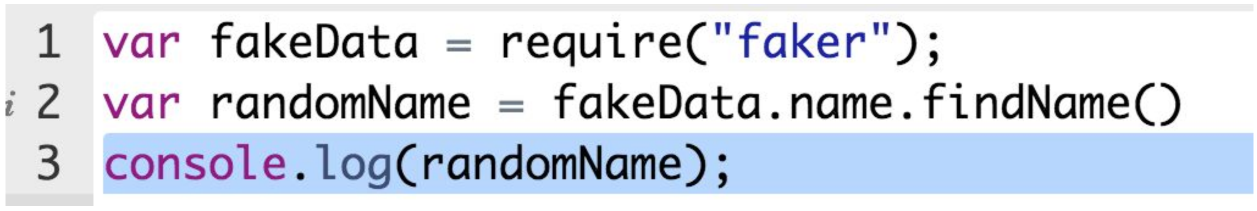
   var faker = require('faker');

   Imports the actual package and store it into a variable called "faker". It can be any variable. It could have been "fakeData", however, if you use a different name, the second line of code should also keep using the new name.

   For instance, let's have:

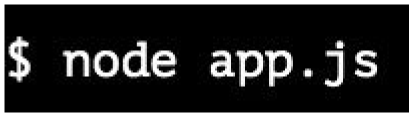
   var *fakeData* = require('faker');
   var randomName = *fakeData*.name.findName();

4. Use "console.log" to display the random name generated by the library. Your code should look something like:



```
var fakeData = require("faker");
var randomName = fakeData.name.findName()
console.log(randomName);
```

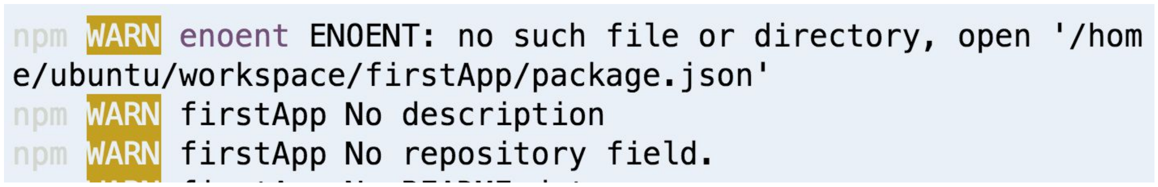
5. Run the app and you should see a new random fake name every time you run it.



```
$ node app.js
```

5. Creating the package.json file

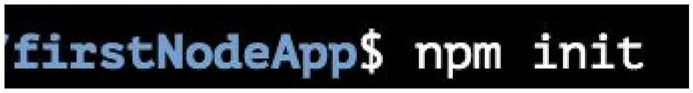
When you installed the faker package, most likely you got a warning message like this:



```
npm WARN enoent ENOENT: no such file or directory, open '/hom
e/ubuntu/workspace/firstApp/package.json'
npm WARN firstApp No description
npm WARN firstApp No repository field.
```

This is because all node apps must have a package.json file. This file includes metadata about the app, such as author, version, description, license, and more importantly all different dependencies (all other node packages) that it uses. Ideally, this file should be created before installing any package, but node is flexible enough and allows us to create it afterwards.

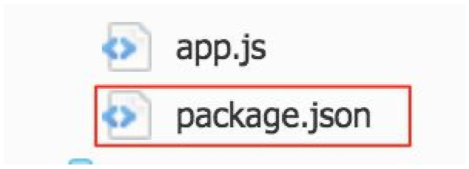
1. Within the "**firstNodeApp**" folder, type:

   ```
   firstNodeApp$ npm init
   ```

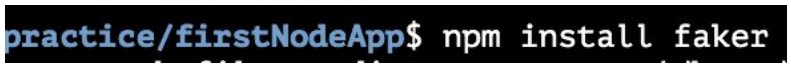
   It will run an interactive utility that will ask you for the basic information to be included in the package.json file. Keep in mind that the app name should be in lowercase.
   To use the default values, just hit enter.

2. Once the "npm init" utility finishes, it will create a package.json file within the "firstNodeApp" folder. Double click on it to open it and see its content. You'll notice that it's a file in JSON (JavaScript Object Notation)



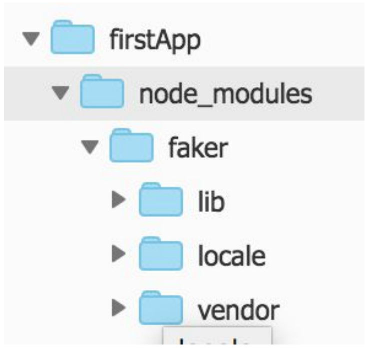
format.

3. After creating the package.json file, reinstall the faker package once more by running:



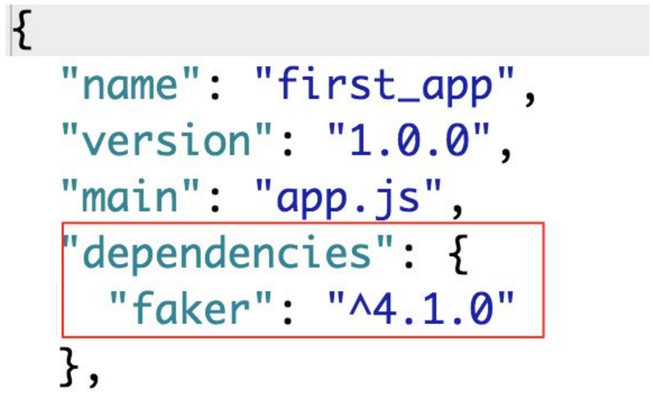
```
practice/firstNodeApp$ npm install faker
```

This should have created a folder called "**node_modules**" and within it, the "**faker**" module. The "node_modules" folder is created only once when installing the first module. Any new modules will be stored within this folder.



Note: A shortcut to install packages is using "i" instead of "install": npm i *package_name*

4. Open the package.json file and notice that the dependency for the "faker" package was added:



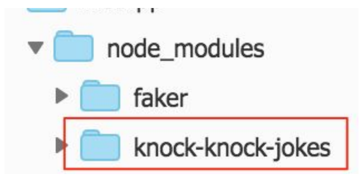
```
{
  "name": "first_app",
  "version": "1.0.0",
  "main": "app.js",
  "dependencies": {
    "faker": "^4.1.0"
  },
```

6. Installing another Node Package

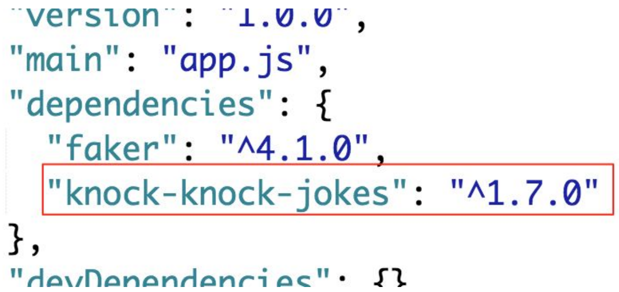
Let's add one more node package but this time using the "--save" parameter.

1. Using the Terminal, within the "**firstNodeApp**" folder type:
   `npm i knock-knock-jokes`

2. Confirm that the "knock-knock-jokes" folder was added within the "node_modules" folder.



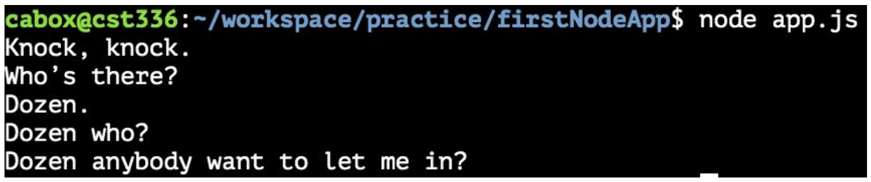
3. Open the package.json file, and confirm that the corresponding dependency was added:



```
  "version": "1.0.0",
  "main": "app.js",
  "dependencies": {
    "faker": "^4.1.0",
    "knock-knock-jokes": "^1.7.0"
  },
  "devDependencies": {}
```

4. Open the **app.js** file and modify it to display a random knock-knock joke. The package documentation is located at: https://www.npmjs.com/package/knock-knock-jokes

Your output should look something like this:



```
cabox@cst336:~/workspace/practice/firstNodeApp$ node app.js
Knock, knock.
Who's there?
Dozen.
Dozen who?
Dozen anybody want to let me in?
```

Review

When installing a new node app:

1. Create a new folder
2. Run “npm init” to create the package.json file
3. Install as many npm packages as needed by running:

   “npm install *package_name*”

   Note: You could use just "i" instead of "install"
4. Create an “app.js” file
5. Import the packages by using something like:

   const variable_name = require("*package_name*");
6. Refer to the package documentation to use it properly