Using the Node Package Manager (npm)

Ramiro Gonzalez

February 01, 2019

At the end of this project one should be able install, update, and remove packages using npm (node package manager). You should be able to understand how to reuse code, install packages using npm, and use those packages.

Background

We will be working with the Linux Command Line and VIM or NANO (text editor). It is assumed that you are running a Ubuntu Distro (Linux Distribution) and have installed node. Reusing code is important in developing programs and npm provides tools and packages; functionalities that have been created by other developers. We assume you are able to get around the Linux file system. Inside the same folder do the following. The version of node used here is v8.11.4

Material

- ☐ Computer
- ☐ Internet Connection
- ☐ Ubuntu Linux (Linux Distribution)

1 Tasks

Complete the following tasks. Note that different operating systems may require different steps.

Task 1

This task will introduce the concept of code reuses. Create a module (small units of independent, reusable code.) and access it through other programs (reuse).

☐ Create a module

1. Create a JavaScript file named moduleCreate.js . The touch command only creates the file and nothing more.

```
$ touch moduleCreate.js
```

2. Edit the file moduleCreate.js. Vim opens the file for editing and also creates the file if it does not exist.

```
$ vim moduleCreate.js
```

Add the following lines to moduleCreate.js

```
exports.myText = "You did it!"
```

☐ Reuse module.

1. Create a JavaScritp file named moduleUse.js

```
$ touch moduleUse.js
```

2. Edit file moduleUse.js

```
$ vim moduleUSe.js
```

3. Add the following lines to moduleUse.js

```
var myModule = require("./moduleCreate.js
    ")
console.log(myModule.myText)
```

4. Run moduleUse.js

```
$ node moduleUse.js
```

If the output is "You did it" then this task has been succesffuly completed.

Task 2

Creating your own modules has its limitations. Developers around the world have created their modules and packaged them. NPM (node package manager) is used to install packages, such packages are a collection of useful modules. We will install the popular packages. This link provides a list of the most popular packages https://www.npmjs.com/browse/depended.

- ☐ Make sure to update before installing anything

```
$ sudo apt update
```

- ☐ Installing lodash

```
$sudo npm install -g lodash
```

  A new directory will be created named *node_modules* it will contain the directory *lodash*. Inside *lodash* you will find JavaScript files, this files contain features and functionalities of lodash.

- ☐ Using lodash. Read about lodash https://lodash.com/
  Create a program that returns a random number between 50 and 100, name it demonstration.js

- ☐ Installing nodemon. -g stands for global, this makes nodemon accessible throughout the system

```
$ sudo npm install -g nodemon
```

  Nodemon is useful for restarting application. https://nodemon.io/

- ☐ Using lodash

```
$ nodemon demonstration.js
```

  Typing 'rs' on the promp will restart the program.
  To exit nodemon use "CTRL-C"

Task 3

In this task you will create a json file. "A JSON file is a file that stores simple data structures and objects in JavaScript Object Notation (JSON) format,

which is a standard data interchange format." https://fileinfo.com/extension/json This is useful when distributing the application. If we want to distribute our application we need to specify the dependencies, that is the packages that were used.

☐ Generate a package.json file

```
$ npm init
```

After running this command you will prompted to add package name, version, description, and much more. You may use the defaults or customize it.

☐ Explore the package.json file

```
$ vim package.json
```

This file contains a list of dependencies.

```
"name": "playground",
"version": "1.0.01",
description: ""
"main": "demonstration.js",
"dependencies" : {
    "lodash": "^4.17.11"
},
devDependencies" : "{},
.............
.............
```

package.json

Task 4

This task will show you how to read and write files using the File System 'fs' module. https://nodejs.org/api/fs.html

☐ Create a json file named sampleData.json and add the following data.

```
{
    "message": "You did it!"
}
```

sampleData.json

An object is created. This object has one **property** named "message".

☐ Create a javaScript file named readData.js and add the following code.

```
1    var fs = require('fs')
2    var data = require('./sampleData.json')
3    console.log(data)
4    fs.readFile('./sampleData.json','utf-8',(err,data) =>
         {
5        console.log(data)
6    })
```

☐ Run readData.js using nodemon

```
1    $ nodemon readData.js
```

Summary

It is important to understand how to use modules. Refer to the following links https://nodejs.org/en/docs/, https://www.w3schools.com/nodejs, https://www.npmjs.com/browse/depended. Different projects require different functionalities and creating your own modules is inefficient. Reusing code can decrease development time and errors.

- Continue learning javaScript
- Use the command line as much as possible
- Look up npm packages that may be of use
- Do not reinvent the wheel, use preexisting modules