**CHAPTER 1**

1. What is node?
   1. javaScript without a browser
   2. javaScript engine
2. Node installation options
   1. Pre-compiled binary
   2. **Version manager** (Do not know how to do this on windows, only choice for Linux)
   3. Compiling from source
3. CLI
   1. Windows CLI
   2. **Git CLI** – (preferred because it understands git, node & linux)
   3. Node CLI

**CHAPTER 2- understanding node** (Important, Examples)

1. javaScript objects in browser vs node
   1. browser
      1. window
      2. location
      3. document
   2. node (removes objects which doesn’t make sense outside browser; it adds objects which helps

Working with modules)

* + 1. global – require
    2. module
    3. process
  1. object available in both environments
     1. console

1. Understanding **require ()** , **module.exports() = \_\_\_**
   1. **require (‘./relay’)** will look for two things
      1. Look for relay.js in current folder
      2. Look for relay folder in current folder and if it exists look for index.js in relay folder
   2. Concepts and errors
      1. require('http’) – global module available via node notice there is no **./**
      2. ‘./’ or ‘’ current directory
      3. ‘../’ parent directory
      4. Missing ‘s’ error

module.exports = exports – by default export assumes that is module.export

* 1. Exports.softly = module.exports.softly
     1. Export assumes that by default it is module.export

1. Understanding call backs in the event loop
   1. When node program start event loop starts with it

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| app.js | public | Routes | | | | | | | | | | |  |  |
| Js | index.js | | | | require(‘./user’) | | | | require(‘./event’) | | |
| HTML | user.js | GET | | POST | | | DELETE | | | UPDATE | |
| CSS | event.js | | GET | | | POST | | DELETE | | | UPDATE |
|  |  | | | | | | | | | | |

**Express App**

Architecture Principle 1

**Application Style**

server --> app --> route/index --> route handling functions

server.js (some people calls it app.js)

* entry file
* node server
* define and start a server in this file
* get all the data in this file; var data = require(‘./data’);
* pass the data to app.js file; var app = require(‘./app’)(data);

app.js

* get data from server
  + module.exports = function(data){
    - pass data to router/index.js;
    - var routes = require(‘./routes’)(flights);
  + return app;
* };

router/index.js

* get data from app
* call handler functions

**Testing Style**

Create a test folder

* Create helper, test subfolders
* In helper create an instance of an original app and run it against data from data folder
* While installing modules for testing use flag --save-dev
* Create a JS file containing all the unit tests

Misc. learning