Section 18-19: Node.js

# Command Line

|  |  |
| --- | --- |
| **Command** | **Description** |
| node  node *fileName* | To access REPL  Run a javascript file from command line |
| npm init | Initialize NPM |
| npm install *packageName* | Install a package from NPM |
| .exit  Ctrl+C | To exit |

# Javascript Code

|  |  |
| --- | --- |
| Code | Description |
| require(“*packageName*”) | Indicates that variable requires a certain package |
| *app*.listen(portN*umber*)  *app*.listen(*portNumber*, function(){ *code* }); | Server listens to http requests on specified port number. Adding a function will cause code to execute while listening. |
| app.use(*variable*); | Necessary to use things such as body-parser |

# Node.js REPL

Node.js is used to connect javascript to the backend.

**REPL** (**R**ead **E**valuation **P**rint **L**oop) allows you toe execute byte-sized chunks of code.

To access it, all you have to is type “node” in the command line. You can tell it worked when a carrot > appears on the left next to where you type your commands. This will allow you to type in code and execute it immediately.

To exit: type “.exit” or do Ctrl+C

# Node.js Express

**Express** is a node.js framework. It can be downloaded using VPM. To download, type “npm install express”. This should add it to the dependencies in the package.json file.

To use it, do the following:

const express = requires(“express”);

const app = express();

You can check a host by typing “localhost:*portNumber*” into your browser.

# GET request

A GET request is when the browser tries to get in touch with the server. It looks like this in javascript:

app.get("/", function(request, response){

*code*

});

* The first parameter is the route.
* The second parameter is the callback function, which tells the server what to do when the request happens. The function’s parameters are “request” and “response”
* The “request” parameter gives you info about the request that was made
* The “response” parameter is how you want to respond to the request. This must be altered.

Response methods

|  |  |
| --- | --- |
| response.send() | Basic response |
| response.send(*filePath*)  response.send(\_\_dirname + “/*filename*.html); | Respond with an HTML file  \_\_dirname gets the path of the current file. This is useful for when other computers try to access a file, as the path may not be the same as the one on your computer. |

# POST request

A POST requests is when the server tries to get data from the browser. To process a POST request, it looks like this in javascript

# *Installations*

* Install NPM
* Install Express: npm install express.
* Install **nodemon** to automatically refresh server when changes are made to code: npm install -g nodemon
  + To add it to a server, cd to correct directory, then in command line type “nodemon *serverFileName*”.
* Install body parser to parse information from a POST request: rpm install body-parser
  + Must add const bodyParser = require(“body-parser”);