#### **Introduction to Installing and Setup of Node**

This section is a fast paced quick intro to NodeJS, and how you start using NodeJS. We discuss what NodeJS is and that it is not a programming language like JavaScript is but an interpreter and Environment for JavaScript.

Node is a lightweight framework developed on Chrome’s V8 engine that can be used for largeScale application development. It's scalable because the server can respond in a non-blocking way.

Process object which is a global that provides information, and control over, the current Node.js process. Because its global it is always available to Node.js applications without using require()

**npm** registry contains packages, many of which are also Node modules, or contain Node module

Demo how to setup your development environment setting up Editor

1. Visual Studio Code as an Editor with ready terminal
2. Creating your first Node File and running the code
3. How Node Works and examples of simple node syntax
4. Where you can find out more about Node with more examples.
5. How to debug and output content into the terminal various command line and console outputs
6. NodeJS programming language environment **REPL** Read Evaluate Print Loop
7. Examples of Node
8. Process Object - with examples
9. What Node Packages and NPM

#### **1 What is Node Setup Dev Environment**

* Setup and introduction to getting started with Node Node website<https://nodejs.org/en/download/>
* Editor Visual Studio Code<https://code.visualstudio.com/>
* NPM - setup<https://www.npmjs.com/>
* Terminal Command+Space or Windows<https://www.microsoft.com/en-ca/p/windows-terminal-preview/>
* Check Node Version and NPM version node -v and npm -v

#### **2 Visual Studio Code Terminal Ready**

* How to use Visual Studio Code
* Open and access Terminal in Visual Studio code
* Editor VIsual Studio Code<https://code.visualstudio.com/>
* Setup JS file for Coding

#### **3 Create a Node file and Run it**

* Create a first js file
* How to run a node js file
* Terminal setup
* Console.log in browser and in terminal
* Use of console.log to output content into console
* ls or dir to list files within the current directory using the terminal
* node app.js running a node file.

#### **4 What is Node and how it Works**

* Node code and how it runs on your machine
* V8 engine
* Asynchronous and non Blocking
* Benefits of Node
* Why node is a good choice
* Node applications.

#### **5 NodeJS resources**

* V8 Chrome and Node
* History of Node
* Browser document object DOM
* Selecting page element
* Window Object in browser
* No window object in Node
* Node global object
* Non Blocking Asynchronous
* Speed difference

#### **6 Working with Node JS**

* CTRL+C to exit from node application
* Run Node Application node appNAME.js
* Create console.log message
* Create Loop - stop loop with CTRL+C
* Process object within node - process.exit()
* for loop and using exit to break loop

console.log(process);

for(let x=0;x<1;x++){

console.log(x);

if(x==30000){

process.exit();

}

}

#### **7 Node JS REPL**

* Type node in the terminal
* To get help .help in REPL mode
* REPL also known as Read Evaluate Print Loop is a programming language environment
* console. Press tab to get a list of available functions.
* global. +tab to get list of available functions within global object
* console option in REPL
* Global Object in Node
* Math in the terminal

#### **8 Common Node Examples**

* Filename and dirname
* Locate file location to use within node application
* Use of node documentation for examples and code samples
* Node global object
* \_\_dirname get directory of current file
* \_\_filename get filename of current file

#### **9 Node Process Object**

* The process object is a global that provides information about, and control over, the current Node.js process
* Process argv - output and use process values within node application
* process.pid
* Logging process argv by index using console.log
* template literal for strings ${} backticks `

console.log(\_\_dirname);

console.log(\_\_filename);

console.log(process.argv);

console.log(process.argv[1]);

const first = process.argv[2];

const second = process.argv[3];

let message = `Hi, welcome ${first} ${second}`;

console.log(message);

#### **10. Use Node Packages NodeMon**

* nodemon is a tool that helps develop node.js based applications by automatically restarting the node application when file changes in the directory are detected.
* Search at https://www.npmjs.com/
* How to install packages
* npm install -g nodemon
* nodemon app.js
* Use of sudo prefix for admin on Mac
* -g global flag

#### **11. Command Line and Console**

* Type code directly within REPL mode
* Use of Console.log to debug and output to terminal console
* Console API options https://nodejs.org/dist/latest/docs/api/console.html
* Log, warn, info, table and count
* console.log %s use of string placeholder
* console.count()
* console.clear()
* console.table()
* console.warn()
* console.info()

#### **Assignment Section #1**

1. Open the REPL mode and list the process object. Return the process.ppid and the pid into the terminal.
2. Run the node script with 2 arguments: your first name and lastname. Create a node application and output process.argv[2] process.argv[3] into the terminal like the example below.

const a = 4;

const b = 6;

console.log(a,b,a+b);

console.log('%s',a);

console.log('Hi Laurence');

console.count('test'+a);

console.count('test'+a);

console.count('test'+a);

console.clear();

console.log('Hi Laurence');

console.warn('warn');

console.info('info');

console.table([{a:1,b:2},{a:5,b:10}]);