**NODE.JS**

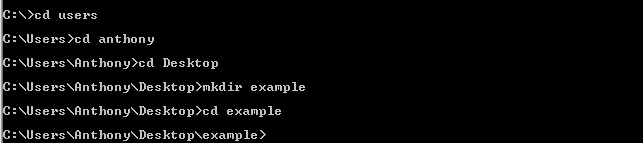
Node.js is the server side JavaScript and is composed of the MEAN stack: **M**ongoDB (document oriented database), **E**xpress Shares (server side framework), **A**ngular.js (client side framework) and **N**ode.js, which is built on the same engine as Chromium.

Node.js currently have two versions with v 10.0.0 having a long-term support. This version can be used to create desktop applications. Node.js also comes with npm, an installer for node.

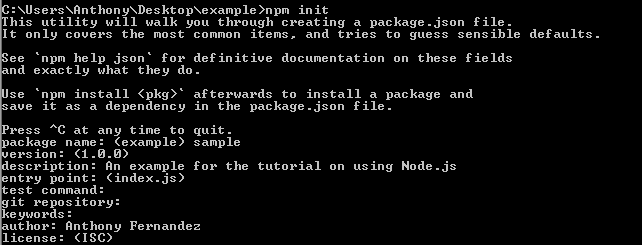
**CREATING A WEB APP USING NODE.JS (Basics)**

*Install the dependencies needed for the project.*

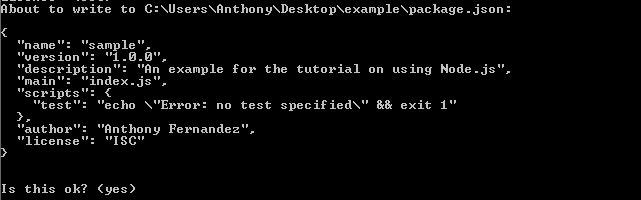
1. Using the command line, create a new directory on whichever place you prefer and change directory there.



1. Run npm init next. This would contain all the details about the app and would generate a package.json file which all dependencies information will be placed.



You will be asked to enter details to be placed on package.json, afterwards the contents will be shown to you and you’ll be asked again if the contents is alright with you. You can always go back to edit it again.



1. Run copy con index.js. This will allow you to create the file index.js which will be the file you’re going to run for the sample.

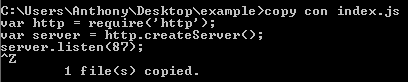
*Basic Scaffolding of the project*

1. After running copy con index.js, enter the following:

var http = require(‘http’);

var server = http.createServer();

server.listen(87);



The following allows us to create a server that runs on our desired port: 87 (changeable).

1. To run the server key in, node index.js. Then open your favorite browser and key in localhost:<specified port number>. As you can see on the screenshot, it is already connected to the server but since index.js is still not filled up with contents, it won’t be able to fetch anything.



1. Let’s go back to our file index.js and add the following:

function handler(request, response) {

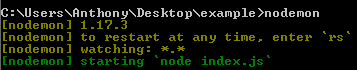
response.end("Hello World!");

}

Placing the handler allows us to send a request to the server and return Hello World as a response.

*Additional Dependencies*

1. Since the actual server doesn’t exactly show a status and you need to stop and start the server every time you make a change, you can install some dependency that can help you show such stats. Given that run: npm install -g nodemon. Nodemon shows you header requests and overall status to allow you to see the resource that is being rendered by visiting the address on your browser.
2. To run your server using nodemon, just key in nodemon. As you can see on the results it will show you the resources it’s getting and starting. It will also automatically detect your changes and would restart the server for you.



*Database connection*

1. In your index.js, add the following on your function handler:

var mysql = require('mysql');

var connection = mysql.createConnection(

{

'host': 'localhost',

'user': 'root',

'password': '',

'database': '<name of database>'

}

);

connection.connect();

This allows you to create a connection to your database. While the next lines will help you in querying from the database.

connection.query('SELECT \* from < table name >',

*Express.js Usage*

Express.js is a backend app framework that facilitates the hard parts to implement in Node.js. To install this framework, key in your cmd: npm install express.js –save.

This allows you to generate HTML files in a pug format in which you don’t need to place opening and closing tags as long as you placed everything in a hierarchical format. It will then be generated as HTML files with proper taggings.