MongoDB, Express JS, Angular 3, Node.js

|  |
| --- |
| JSON , javascript object notation.   * Json is a lightweight data exchange format, * Language Independent (uses JavaScript syntax but the Json format is only text like xml) * Self describing like xml. * Does not use end tag like xml * shorter to write * quicker to read * uses arrays * json can be parsed by Javascript functions |

|  |
| --- |
| JSON syntax  Json syntax is derived from JavaScript syntax  Rules:   * Data in Name/value pairs * Data separated by commas * Objects are enclosed in curly braces * Arrays are housed in square brackets. * E.g of Name value pair: “firstName” : “manoj”   E.g of and Object {“firstName” : “manoj” }  if we have two values we will use commas  e.g. {“firstName” : “manoj” , “lastName” : “N”}  ------------------------------------------------------------------------------------  Arrays example with 3 values:  “Staff” : [ {“firstName” : “Mike” , “lastName” : “Ross”} ,  {“firstName” : “Jack” , “lastName” : “Doe”} ,  {“firstName” : “Jane” , “lastName” : “James”}  ] |

All the Json document start and end with a curly brace { }

“Key” : “value” ,

|  |
| --- |
| What is Node.js  Runtime environment for server-side & n.wing apps.  Open source  Cross platform  Built on top of chrome’s javascript V8 engine  Node apps are written in JavaScript  Lightweight  We can write both frontend & backend apps in Same language |

Companies like : Yahoo, Netflix, Paypal, Uber, Ebay, Google, GitHub, Walmart use Node in some aspect of their software.

|  |
| --- |
| Nodes Package Manager  a module is also known as a Library.  Online repo for publishing open source node.js projects.  Cmd utility for installing & interacting with packages & modules  Used to manage packages  Makes it easy for developers to Update, share & reuse code  Comes installed with Node.js  *npm --version*  to check version info  *npm install npm@latest -g*  globally |

|  |
| --- |
| Npm basic commands:  npm install <package name > -g  npm uninstall <package name > -g  npm search <package name >  npm ls -g  npm update -g |

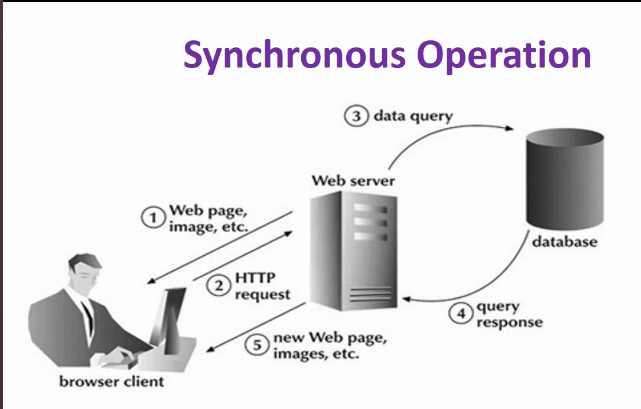
**Global objects**

|  |  |
| --- | --- |
| Class:Buffer | Used to handle binary data. |
| \_\_dirname | Dir name of current module |
| \_\_filename | File name of current module. |
| clearImmediate(immediateObject) | Cancels and immediate object. |
| console | Prints to console |
| exports | A reference to the module.exports |
| require() | To require modules |

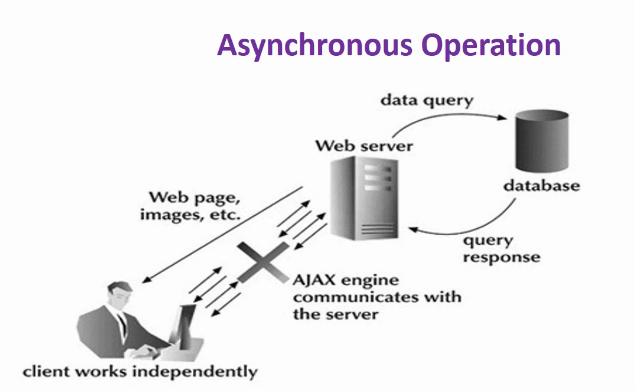
==============

Understanding Synchronous & Asynchronous Tasks

* Sync task blocks a process till the previous task completes.
* Async task is a Non-Blocking & can start any task, this avoids blocking

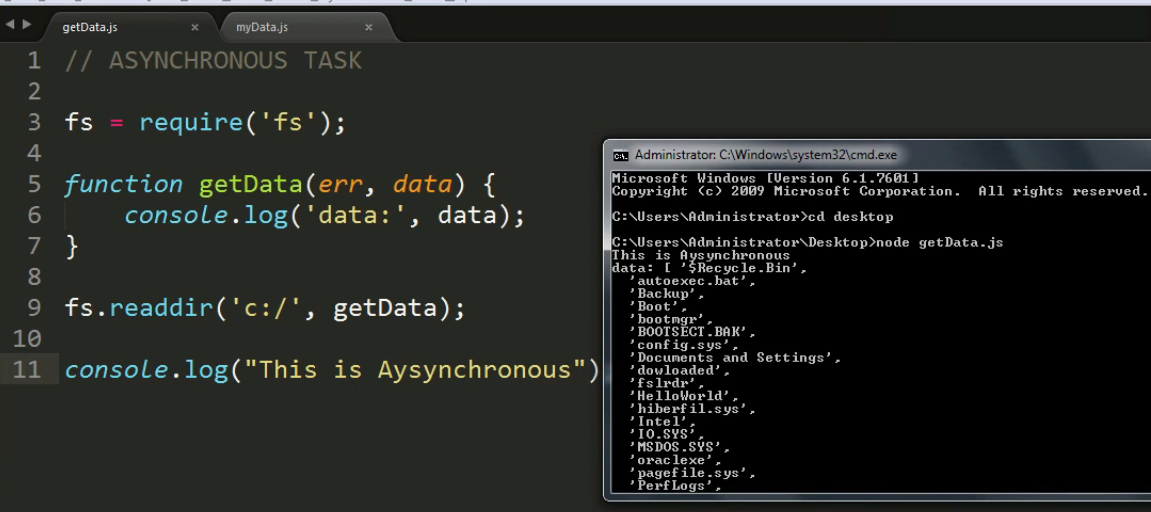


------------------------------------------------------- -



=====================

An Example of Async process:



fs is filesystem.

fs.reader that will take 2 parameters the err and data & will read the dir in c: and get the data and also give a log on console.

We have a log on the bottom , but it executes first due to the Async nature of the task

It didnot wait for the previous task to complete.

==========

Example for Synchronous task.

|  |
| --- |
| fs = require(‘fs’);  data = fs.readdirSync(‘c:/’);  console.log(‘data:’ , data);  console.log(“This is Synchronous”); |

*save the file as getData.js*

*run using* node getData.js

=============

code.visualstudio.com

=============

|  |
| --- |
| What is MongoDB   * A document database * Stores data in flexible, Json like documents. the fields ca vary from doc to doc and data structure can be changed from time to time * The document model maps to the objects in your app code making data easy to yse * Is free & open-source * A distributed database, horizontal scaling and geographic distribution which are built in and easy to use. * Is flexible & scalable with querying & indexing so we can do ad-hoc queries, indexing and real time aggregation which provides powerfull ways to analyse our data. |

Download MongoDB

create the path variables

create a /data/db folder to store the database

open 2 terminal shells

mongod => deamon

mongo => client shell to interact with the db

^c to terminal both

MongoDB datatypes

String

Integer

Boolean

Arrays

Timestamp

Object Used for embedded documents

Null Stores null value

Date

Object ID Stores Object ID

Binary data Stores Binary data

code Stores Javascript code

Regular expression

Mongo db VS Rdbms

==============================

CREATING A MEAN STACK APPLICATION,

SHOPPING CART

create 3 directories

shoppingList =>ServerCode & clientCode

Then initialise the project

we need a package.json file to keep your dependencies

Any project using Node.js will need a package.json file

It is structured in Json format

create a package.json using **npm init**

we can also seperate the dependencies for production and development

**cd** shoppinglist

cd servercode

**npm init**

Project name

initial version

description

entry point (main file) *usually a index.js file*

test command

git directory

keywords

license

or **npm init --yes** for instant project initialization

=======

Instal dependencies

we will need: EXPRESS, CORS, BODY-PARSER, MONGOOSE

Express:

is a minimal and flexible Node.js web application framework

Provides robust features for web and mobile applications

Installed with npm

**npm install espress --save** *save in the package.json file*

CORS

Cross-origin resource sharing

is a Node.js package for providing a Connect/Express middleware

allows your web app to make calls to your web API server

**npm install cors --save**

Body-parser

to parse json related data

Extract the entire body portion of an incoming request stream & exposes it on req.body.

**npm install body-parser --save**

**npm install monsoose --save**

================================

in the servercode dir make a file called **index.js**

inside index.js

|  |
| --- |
| var express = require(‘express’);  var mongoose = require(‘mongoose’);  var bodyparser = require(‘body-parser’);  var cors = require(‘cors’);  var app = express(); *creating an instance of express* |

============================

in servercode dir **mongod** command to start the deamon and let it run see the **port**

now to connect with mongoose

|  |
| --- |
| var express = require(‘express’);  var mongoose = require(‘mongoose’);  var bodyparser = require(‘body-parser’);  var cors = require(‘cors’);  var app = express(); *creating an instance of express*  mongoose.connect(‘mongod://localhost:27017/shoppinglist’);  mongoose.connection.on(‘connected’, () => {  console.log(‘CONNECTED TO MONGODB’);  });  mongoose.connection.on(‘error’, (err) => {  console.log(‘err’);  } ));  constant PORT = 3000;  app.listen(PORT, () => {  console.log(‘Server started at port : ‘+PORT‘);  }); |

cmd

node index.js

==========================

Installing Node monitor

we dont need to restart server again and again the server

**nodemon**

npm install -g nodemon

*npm install --save-dev nodemon other way, not necessary to run this one also*

**nodemon** cmd to start it.

===============

TESTING A ROUTE

**constant PORT = 3000;** *after this line in the code*

|  |
| --- |
| app.get(‘/’,( req, res) => {  res.send(“Roger you are connected to the server”);  }); |

we should see the message in the browser.

====================================================

CREATE A DIR FOR ROUTES & MODEL SCHEMA

inside servercode dir

mkdir route

add a file called **route.js**

===

inside servercode dir

mkdir model

file called **shoppingItem.js**

===================================

Middleware

7.10