Node.js is a platform built on [Chrome's JavaScript runtime](http://code.google.com/p/v8/) for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

* Asynchronous and Event Driven
* Very Fast Being built on Google Chrome's V8 JavaScript Engine
* Single Threaded but Highly Scalable

[Installing Node.js](https://docs.npmjs.com/getting-started/installing-node#installing-nodejs)

If you're using OS X or Windows, the best way to install Node.js is to use one of the installers from [nodejs.org](https://nodejs.org/).

Test: Run node -v. The version should be higher than v0.10.32.

## [Updating npm](https://docs.npmjs.com/getting-started/installing-node#updating-npm)

npm makes it easy for JavaScript developers to share and reuse code, and it makes it easy to update the code that you're sharing.

Node comes with npm installed so you should have a version of npm. However, npm gets updated more frequently than Node does, so you'll want to make sure it's the latest version.

sudo **npm install npm –g**

## [Installing npm manually](https://docs.npmjs.com/getting-started/installing-node#installing-npm-manually)

For more advanced users.

The npm module is available for download at

[https://registry.npmjs.org/npm/-/npm-{VERSION}.tgz](https://registry.npmjs.org/npm/-/npm-%7bVERSION%7d.tgz).

Node.js modules are one kind of package which can be published to npm. When you create a new module, you want to start with the package.json file.

You can use npm init to create the package.json. It will prompt you for values for the package.json fields. The two required fields are name and version. You'll also want to have a value for main. You can use the default, index.js.

If you want to add information for the author field, you can use the following format (email and web site are both optional):

Your Name <email@example.com> (http://example.com)

Once your package.json file is created, you'll want to create the file that will be loaded when your module is required. If you used the default, this is index.js.

In that file, add a function as a property of the exports object. This will make the function available to other code.

exports.printMsg = function() {

  console.log("This is a message from the demo package");

}

Test:

1. Publish your package to npm
2. Make a new directory outside of your project and cd into it
3. Run npm install <package>
4. Create a test.js file which requires the package and calls the method
5. Run node test.js. The message should be output.

npm registration:

You can publish any directory that has a package.json file, e.g. a [node module](https://docs.npmjs.com/getting-started/creating-node-modules).

## [Creating a user](https://docs.npmjs.com/getting-started/publishing-npm-packages#creating-a-user)

To publish, you must have a user on the npm registry. If you don't have one, create it with npm adduser. If you created one on the site, use npm login to store the credentials on the client.

Test: Use npm config ls to ensure that the credentials are stored on your client. Check that it has been added to the registry by going to <https://npmjs.com/~>.

Install, configure and start guide for mongo db.  
1) Download: http://www.mongodb.org/downloads

Execute .msi file, during installation wizard select custom option and give installation path "d:\mongodb"

2) Create Configuration file inside C:\Bitnami\meanstack-3.2.4-1\mongodb: mongo.config and dbpath &logpath as below.

##store data here  
dbpath=C:\Bitnami\meanstack-3.2.4-1\mongodb\data

##all output go here  
logpath=D:\desktop\Rogers\mongodb\log\mongo.log

##logs read and write operations  
diaglog=3

port = 27017

3) Run MongoDB server:   
d:\mongodb\bin>mongod --config C:\Bitnami\meanstack-3.2.4-1\mongodb\mongo.config  
  
4) Connect to MongoDB  
C:\Bitnami\meanstack-3.2.4-1\mongodb \bin>mongo

4.1 In MongoDB, both database and table are created automatically when the first time data is inserted. Use database-name, to switch to your database (even this is not created yet) but it will be created automatically.  
4.2 In MongoDB, collection means table in SQL.  
4.3 Insert Record: db.tablename.insert({data}) or db.tablename.save({data})  
4.4 Update Record: db.tablename.update({criteria},{$set: {new value}})  
4.5 Find Record: db.tablename.find({criteria})  
4.6 Delete Record: db.tablename.remove({criteria})

> use testdb  
> db.users.insert({username:"Guest",password:"123456"})  
> db.users.find()  
> show collections  
> db.users.update({username:"Guest"},{$set:{password:"hello123"}})  
> db.users.remove({username:"Guest"})