**Introduction**

* What is Node.js ?

Ans : Node.js is not a language. It is an open source server environment.

It is a popular tool for almost any kind of project.

It is mostly used to create API and also use for creating web applications.

* What is REPL in Node.js ?

Ans: NEPL stands for Read-Eval-Print-Loop also known as Node Shell.

It is a quick and easy way to test simple Node.js and JavaScript Code.

You can define variables and perform operations on them.

* What id Module in Node.js ?

Ans : Consider modules to be same as JavaScript libraries.

Module is a set of functions that you want to include in your applications.

Using module files can be re used through the Node.js Applications.

Types of Modules:

1. Core Module : Node.js is a light weight framework. The core modules include bare minimum functionalities of Node.js. These core modules are compiled into its binary distribution and load automatically when Node.js process starts.

Here is some important core modules :

* http
* url
* path
* fs etc..

1. Local Module : Local modules are modules created locally in your Node.js application. These modules include different functionalities of your application in separate files and folders. You can also package it and distribute it via NPM, so that Node.js community can use it.

In Node.js, module should be placed in a separate JavaScript file.

1. Built-in Modules : Node.js has a set of built-in modules which you can use without any further installation.

Here is some important built-in Modules:

* Buffer
* Cluster
* Dns
* Events
* Domain
* http
* https
* readline
* stream
* v8
* ytil etc….
* Node.js is Asynchronous or Synchronous Language ?

Ans : Node.js is Asynchronous language.

In Synchronous, Suppose 4 process is running then one process is waiting for another process it means process 2 is waiting for process 1 to be complete and process 3 is waiting till process 2 completed.

But Node.js is Asynchronous langue. In Asynchronous language, if 4 process is running at same time then no any process will wait for other. If process 1 is pending and process 2 is coming to be execute then process 2 is not waiting for process 1 to be completed and so on…

So Node.js is faster execution ability.

* What is callback ?

Ans : Callback is an asynchronous equivalent for a function. A callback function is called at the completion of a given task. Node makes heavy use of callbacks. All the APIs of Node are written in such a way that they support callbacks.

* What id Chaining in Node.js ?

Ans : Chaining in Node.js can be achieved using the async npm module. In order to install the async module, we need to run the following script in our directory:

npm init

npm i async

There are two most commonly used methods for chaining functions provided by the async module:

* **parallel(tasks, callback):** The tasks is a collection of functions that runs parallel in practice through I/O switching. If any function in the collection tasks returns an error, the callback function is fired. Once all the functions are completed, the data is passed to the callback function as an array. The callback function is optional.
* **series(tasks, callback)**: Each function in tasks run only after the previous function is completed. If any of the functions throw an error, the subsequent functions are not executed and the callback is fired with an error value. On completion of tasks, the data is passed into the callback function as an array.