NodeJS

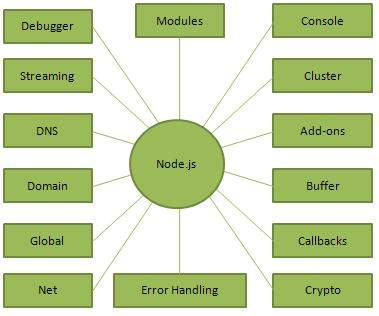
What is NodeJs?

~ NodeJs is a run environment and library for running Java script applications outside the browser

~ Its mostly used to run the real time server applications and shines through the performance using non block I/O and asynchronous events.

~ Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

~  Node.js applications never buffer any data. These applications simply output the data in chunks.



~ Execute js file using node.js

node main.js

~ A Node.js application consists of following three important parts −

* Import required modules − We use require directive to load a Node.js module.
* Create server − A server which will listen to client's request similar to Apache HTTP Server.
* Read request and return response − server created in earlier step will read HTTP request made by client which can be a browser or console and return the response.

~ NPM—Node Package Manger

This is used to install required npm modules

npm install <module\_name> -g(global)

~ 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.

~Although events look quite similar to callbacks, the difference lies in the fact that callback functions are called when an asynchronous function returns its result, whereas event handling works on the observer pattern. The functions that listen to events act as **Observers**

**~**EventEmitter class which are used to bind events and event-listeners

1. emit(event, [arg1], [arg2], [...])

Execute each of the listeners in order with the supplied arguments. Returns true if the event had listeners, false otherwise.

1. on(event, listener)

Adds a listener at the end of the listeners array for the specified event. No checks are made to see if the listener has already been added. Multiple calls passing the same combination of event and listener will result in the listener being added multiple times. Returns emitter, so calls can be chained.