Event loop

Another important thing about Node.js is that it's event driven, and it has a concept called event loop. Node.js works in a single thread. It has a single processor. Below you see the list of tasks. Consider it as a single thread. The tasks will be done in sequence, when the first task is done, the second task will start, when the second task is complete, the third task will start and when the third is finished, the fourth will happen and so forth. It will go in a load.

If the new task comes in, it will be added on the top of the task and it's going to be task 5 and so forth

* Event Loop is a single threaded and semi-infinite loop.
* The reason why this is called a semi-infinite loop is because this actually quits at some point when there is no more work to be done.
* If there is work and  there is no timer is set for that particular event than loop continues until work is not completed.
* Event loop executes the callback and provide appropriate results. Since, it’s in a loop.. The process simply keeps going.
* The event loop simply iterate over the event queue which is basically a list of events and callbacks of completed operations.
* Node starts the event loop by checking for any expired timers in the timers queue, and go through each queue in each step while maintaining a reference counter of total items to be processed.
* After processing the close handlers queue, if there are no items to be processed in any queue, the loop will exit.
* Processing of each queue in the event loop can be considered as a phase of the event loop.

Features

**Real Time Communication**Real time web solutions are web technologies which enable clients to achieve and receive information as soon as they are published by their authors instead of checking by software or refreshing pages by users manually whether there is new data to expose or not.  
  
**Cross Platform**Cross platform means that Nodejs is almost independent from platform and is enabled to be installed on every machine with different operating systems such as Windows, Linux, Macintosh. So Nodejs does not need specific preparation and it is portable.   
  
**Server Side Application**There are situations where it is not possible to give all responsibilities to the client due to the fact that the client does not have the ability to accept them. For instance, if we want to store data on a database or do some special processing on a program we need to put these kind of functionalities on the server which is remote from the client and the client can try to achieve their purpose.  
  
**Event Driven Architecture**If we are working with some data management systems which have many nodes with different statuses, event driven enables us to find out other statuses very quickly so we are able to respond properly. Think about real estate trading, where houses are for sale or rent and customers want to be aware which house is still available. As soon as a house is selected and is sold its status should be changed for others.  
  
**Asynchronous I/O**There are two approaches for I/O, the simple way is synchronous which blocks resources and progress until communication is completed which causes waiting and wastes a lot of resources especially if we have too many I/O.  
  
Another way is asynchronous, which allows critical operations to do their job while waiting for I/O  
  
**Throughput**In such a scenario which we need to send and receive messages, there are possibilities that some messages cannot reach the stations so the rate of successful messages which are received correctly on one channel is called throughput.   
  
**Scalability**Scalability is a capability of an application that can grow by encountering large scenarios such as large amounts of data or nodes. So unexpected increase of quantity will not decrease quality of system.