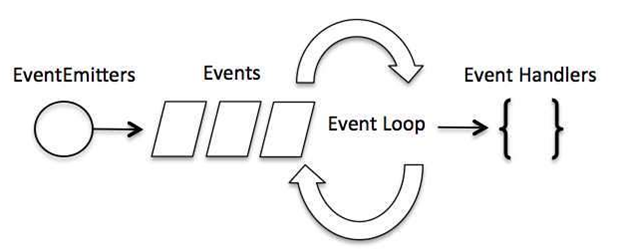
Node.js Events

In Node.js applications, Events and Callbacks concepts are used to provide concurrency. As Node.js applications are single threaded and every API of Node js are asynchronous. So it uses async function to maintain the concurrency. Node uses observer pattern. Node thread keeps an event loop and after the completion of any task, it fires the corresponding event which signals the event listener function to get executed.

Event Driven Programming

Node.js uses event driven programming. It means as soon as Node starts its server, it simply initiates its variables, declares functions and then simply waits for event to occur. It is the one of the reason why Node.js is pretty fast compared to other similar technologies.

There is a main loop in the event driven application that listens for events, and then triggers a callback function when one of those events is detected.



Difference between Events and Callbacks:

Although, Events and Callbacks look similar but the differences lies in the fact that callback functions are called when an asynchronous function returns its result where as event handling works on the observer pattern. Whenever an event gets fired, its listener function starts executing. Node.js has multiple in-built events available through events module and EventEmitter class which is used to bind events and event listeners.

**EventEmitter class to bind event and event listener:**

1. // Import events module
2. var events = require('events');
3. // Create an eventEmitter object
4. var eventEmitter = new events.EventEmitter();

**To bind event handler with an event:**

1. // Bind event and even handler as follows
2. eventEmitter.on('eventName', eventHandler);

**To fire an event:**

1. // Fire an event
2. eventEmitter.emit('eventName');