**Node.js Events and EventEmitter**

One of the reasons for Node.js’ high speed is the fact that it is coded around events. Instead of reading all the files necessary with every request (like PHP), with Node you just start your server, initiate most of the variables, declare your functions and then just wait for an event to occur.

First I will show you how normal events are emitted. As event I will take the example that when a person enters a shop, a bell may ring to show his presence.  It will act a bit like the **observer pattern** where our event acts like subject and all functions attached to the event are like observers. So the shop example:

var events = require('events');

var eventEmitter = new events.EventEmitter();

var ringBell = function ringBell()

{

console.log('ring ring ring');

}

eventEmitter.on('doorOpen', ringBell);

eventEmitter.emit('doorOpen');

* First we load the **events module**, which is part of the Node.js core
* Then we create a new instance of the **EventEmitterclass** (which we are going to extend later).
* After then we place the**ringBell**function inside a variable so that it can be called that way. It simply prints ‘ring ring ring’ in our console.

Try …adding few more listeners to ‘doorOpen’ event .

**eventEmitter.on(‘doorOpen’, doSomething);**

**eventEmitter.on(‘doorOpen’, doSomethingElse);**

eventEmitter.emit('doorOpen');

**eventEmitter.listeners('doorOpen')**

**eventEmitter.listenerCount()**

**emitter.removeListener(eventName, listener)**

**emitter**.removeAllListeners(‘doorOpen’);

ToDO :

In Node Application, any async function accepts a callback as a last parameter and the callback function accepts error as a first parameter.

var fs = require("fs");

fs.readFile('input.txt', function (err, data) {

if (err){

console.log(err.stack);

return;

}

console.log(data.toString());

});

console.log("Program Ended");