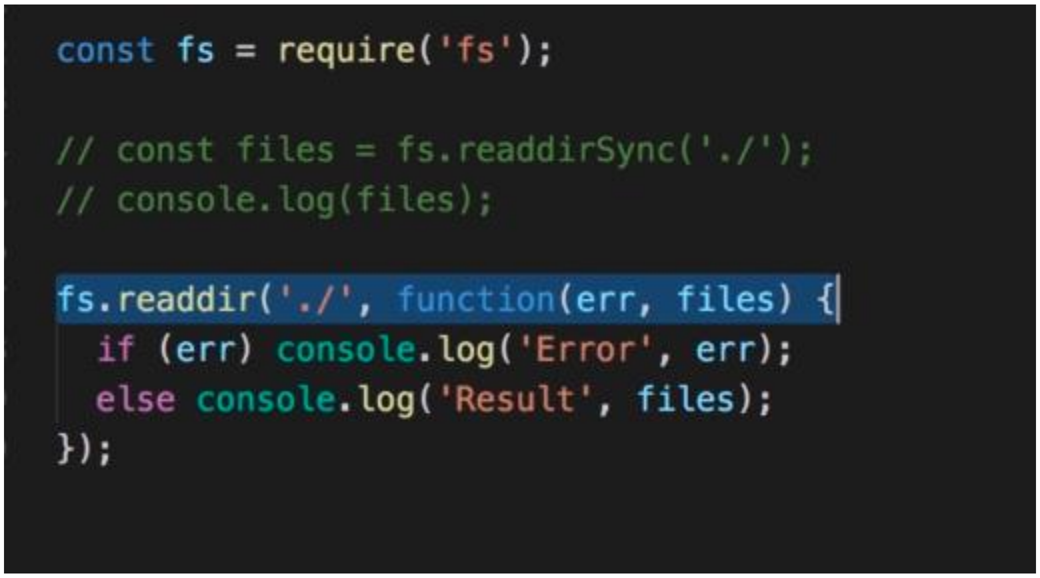
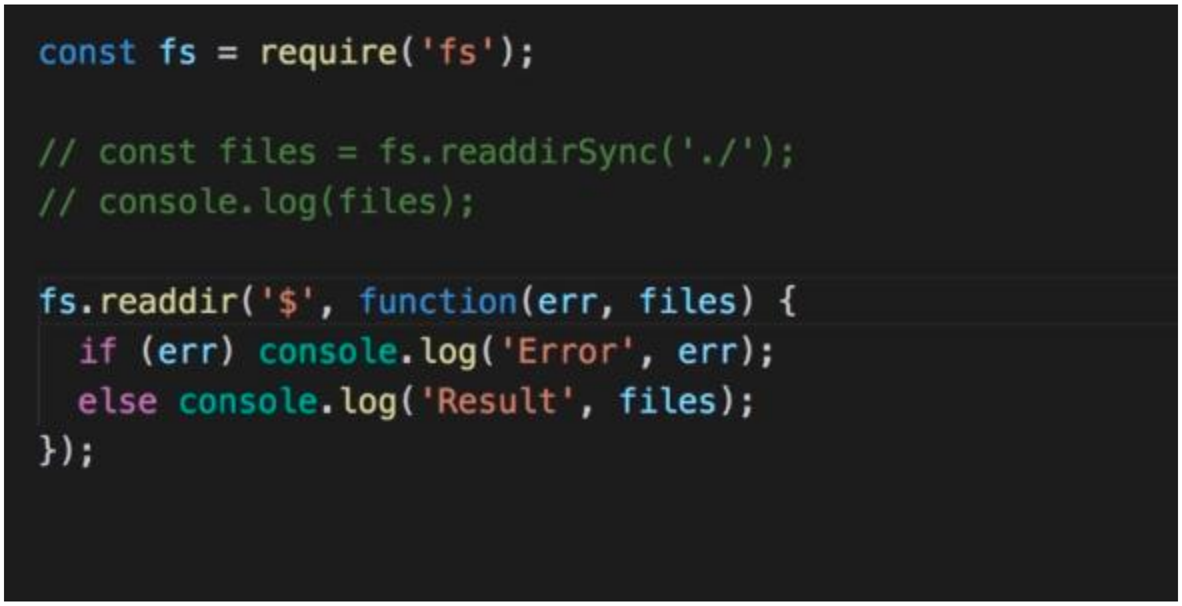
Callback Examples

Reading data asynchronously



```
const fs = require('fs');

// const files = fs.readdirSync('./');
// console.log(files);

fs.readdir('./', function(err, files) {
  if (err) console.log('Error', err);
  else console.log('Result', files);
});
```

To make error in callback:



```
const fs = require('fs');

// const files = fs.readdirSync('./');
// console.log(files);

fs.readdir('$', function(err, files) {
  if (err) console.log('Error', err);
  else console.log('Result', files);
});
```

Event Modules

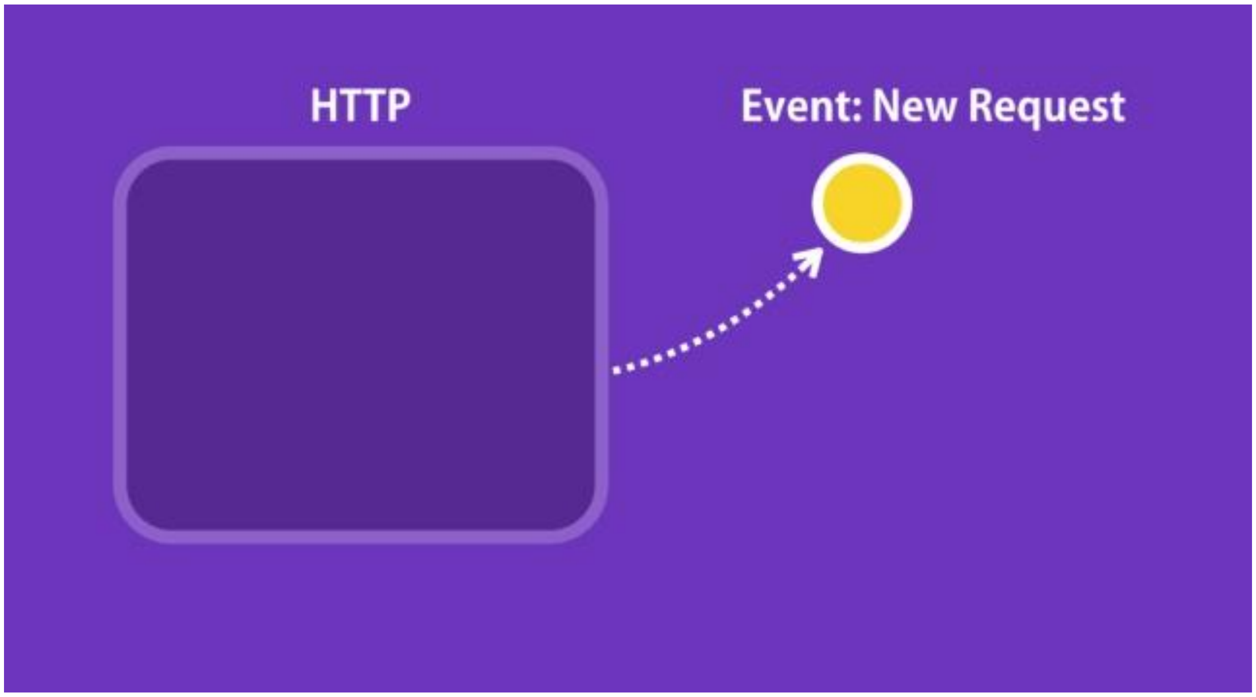
What is an event?

An event is a signal that something has happened.

Example: In Node, we have class HTTP, that is used to build a web server that listens on a port.

Now, whenever we receive a request on that port

What happens?



What is the job of application on receiving that event?

Respond to the event and return response

```
// whenever you will load an event, an EventEmitter class object is returned
// A class contains functions and properties

const EventEmitter = require('events');

// create object of an EventEmitter class to invoke functions
const emitter = new EventEmitter();
```

```
// Register the event

// emitter has bunch of methods
// emit() to raise an event - make a noise in english
emitter.emit('messageLogged');
```

Register the event to be handled

```
// Register a listener
emitter.on('messageLogged', function(){
   console.log('Listener called');
});
```

Passing Arguments to Events

To send multiple values to an event, send as object.

```
// emitter has bunch of methods
// emit() to raise an event - make a noise in english
 emitter.emit('messageLogged', {id: 1, url: 'http://'});
```

Use in the event listener

```
// Register a listener
   emitter.on('messageLogged', function(arg){
     console.log('Listener called', arg);
});
```

Do IT Yourself: Convert function into arrow function in the event listener.

Do It Yourself: Raise an Event for Logging

Extending EventEmitter

Logger.js

```
var url = 'http://mylogger.io/log';

function log(message)
{
    // send an HTTP request
    console.log(message);
}


module.exports = log;
```

To raise an event with arguments from logger:

```

const EventEmitter = require('events');
const emitter = new EventEmitter();

var url = 'http://mylogger.io/log';

function log(message)
{
    // send an HTTP request
    console.log(message);
}


// raise an event
emitter.emit('messageLogged', {id: 1, url: 'http://'});

module.exports = log;
```

Handle the event in testlogger.js

Testlogger.js

```
const EventEmitter = require('events');
const emitter = new EventEmitter();
```

```
// Register a listener
emitter.on('messageLogged', function(arg){
   console.log('Listener called', arg);
});


const log = require ('./logger');
log('message');
```

Check why the event listener is not called in testlogger.js

Answer: There are two separate EventEmitter objects created in both the js files namely, logger,js and testlogger.js

Solution is: Create a class that has all the capabilities of the event emitter.

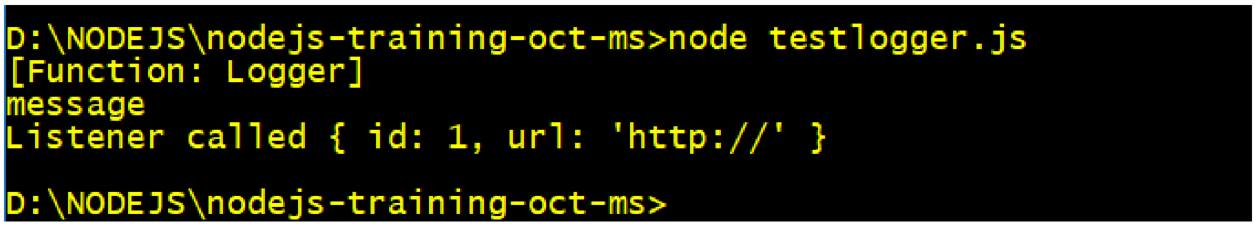
Logger.js

```
const EventEmitter = require('events');

var url = 'http://mylogger.io/log';

// ES6 - sugar syntax for creating constructor function
class  Logger extends EventEmitter{
// a function inside a class is called method
log(message)
  {
    // send an HTTP request
    console.log(message);
   // raise an event from the class
    this.emit('messageLogged', {id: 1, url: 'http://'});
  }
}

module.exports = Logger;
```

Testlogger.js

```
const EventEmitter = require('events')

const LoggerClass = require ('./logger');
console.log(LoggerClass);
const x = new LoggerClass();

// Register a listener
x.on('messageLogged', function(arg){
   console.log('Listener called', arg);
});

x.log('message');
```

Output:



```
D:\NODEJS\nodejs-training-oct-ms>node testlogger.js
[Function: Logger]
message
Listener called { id: 1, url: 'http://' }

D:\NODEJS\nodejs-training-oct-ms>
```

Working with HTTP

Create an HTTP server that listens to port 3000 and serves the request.

Use: http module that contains bunch of classes, properties, methods, and events.

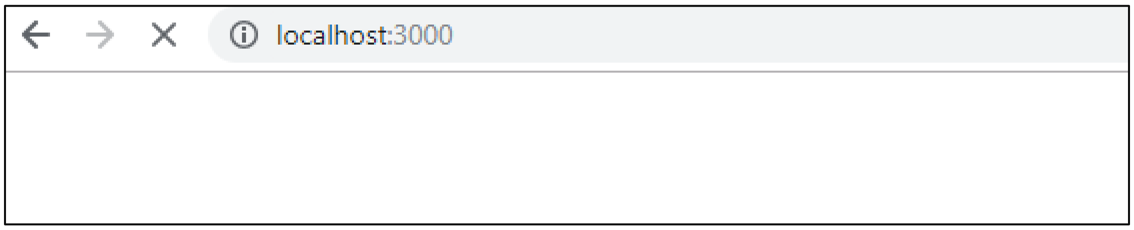
httpserver.js

```
const http = require('http');

// create a web server
// The web server is an event emitter
const server = http.createServer();

// when server is emitting an event, we will provide a handler to handle it
// add the code here to handle the event

server.listen(3000);

console.log("Listening on port 3000");
```

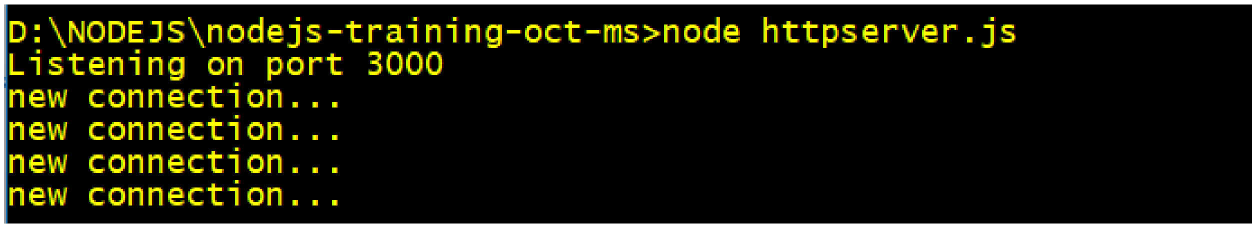
Add the listener:

```
// when server is emitting an event, we will provide a handler to handle it
server.on('connection', function(socket){
   console.log("new connection...");
});
```

Run: Send the request on localhost:3000 and observe console.



Observe CMD

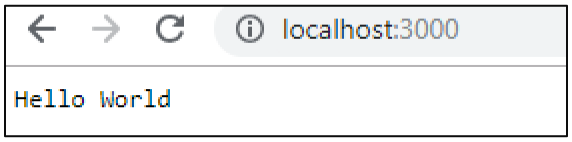


```
D:\NODEJS\nodejs-training-oct-ms>node httpserver.js
Listening on port 3000
new connection...
new connection...
new connection...
new connection...
```

Modify the code to handle callback function on createServer().

```
const http = require('http');

// create a web server
// The web server is an event emitter
const server = http.createServer(function(req, res){
    if(req.url === '/')
    {
       res.write('Hello World');
       res.end();
    }

});

server.listen(3000);

console.log("Listening on port 3000");
```

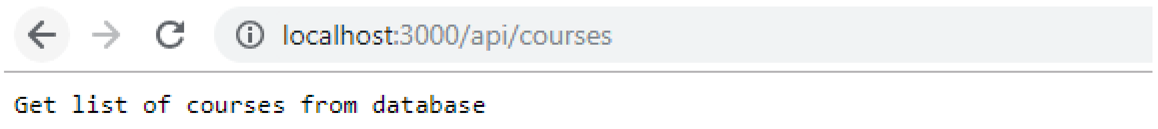
Run:



```
localhost:3000

Hello World
```

Add more route to http.

```
// create a web server
// The web server is an event emitter
const server = http.createServer(function(req, res){
    if(req.url === '/')
    {
       res.write('Hello World');
       res.end();
    }

    if(req.url=== '/api/courses')
    {
        res.write("Get list of courses from database");
        res.end();
    }
});
```



```
localhost:3000/api/courses
```

```
Get list of courses from database
```