SWE 363: Web Engineering & Development

Module 10

Node.js

References

- www.w3schools.com
- www.nodejs.org

Objectives

- ❑ Learn Node.js
- ❑ Syntax of Node.js
- ❑ Modules of Node.js
- ❑ Node.js with database

Outline

- ❑ What is Node.js
- ❑ Install Node.js
- ❑ Modules
- ❑ Events
- ❑ Send an Email
- ❑ Database

What is Node.js?

- Open source server environment.
- Allows you to run JavaScript on the server.
- It is free
- Runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)

- Why Node.js?
  - Node.js uses asynchronous programming!
    - In contrast to PHP or ASP, Node.js eliminates the waiting, and simply continues with the next request.

What Can Node.js Do?

- Generate dynamic page content
- It can create, open, read, write, delete, and close files on the server
- Collect form data
- It can add, delete, modify data in your database

- What is a Node.js File?
  - Node.js files contain tasks that will be executed on certain events
  - A typical event is someone trying to access a port on the server
  - Node.js files must be initiated on the server before having any effect

Node.js for....

- ❑ Web application
- ❑ Websocket server
- ❑ Ad server
- ❑ Proxy server
- ❑ Streaming server
- ❑ Fast file upload client
- ❑ Any Real-time data apps
- ❑ Anything with high I/O

Success Stories…..

Linked in

Rails to Node
« Servers were cut to 3 from 30 »
« Running up to 20x faster in some scenarios »
« Frontend and backend mobile teams could be combined [...] »

PayPal™

Java to Node
« Built almost twice as fast with fewer people »
« Double the requests per second »
« 35% decrease in the average response time »

GROUPON

Walmart

YAHOO!

UBER

Get Started

- ❑ Download Node.js
  - https://nodejs.org
- ❑ Let us create our first Node.js file:
  - Create a Node.js file named *"myfirst.js"*, and add the following code:

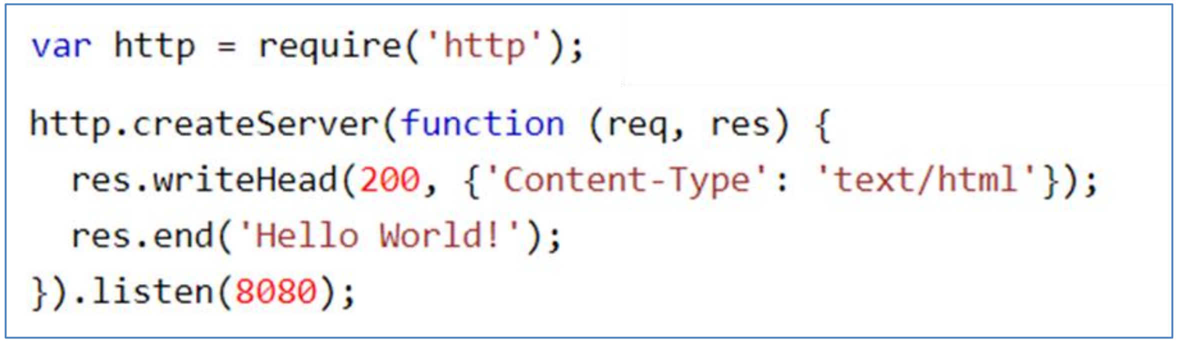
```
var http = require('http');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Hello World!');
}).listen(8080);
```

The code tells the computer to write "Hello World!" if anyone (e.g. a web browser) tries to access your computer on port 8080.

  - Initiate the Node.js File through Command Line Interface
    - ➢ C:\node myfirst.js
  - Now, your computer works as a server! If anyone tries to access your computer on port 8080, they will get a "Hello World!" message in return!
  - Start your internet browser, and type in the address: http://localhost:8080

Node.js Modules

- A set of functions you want to include in your application.
- Consider modules to be the same as JavaScript libraries.
- Built-in Modules
  - To include a module, use the require() function
  - Example:



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

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Hello World!');
}).listen(8080);
```

Node.js Modules…

- <u>User Modules</u>
  - Use the exports keyword to make properties and methods available outside the module file.
  - Example:
    - Create a module that returns the current date and time:

```
exports.myDateTime = function () {
  return Date();
};
```

    - Save the code above in a file called "myfirstmodule.js“
    - Include your own module

```
var http = require('http');
var dt = require('./myfirstmodule');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write("The date and time are currently: " + dt.myDateTime());
  res.end();
}).listen(8080);
```

HTTP Module

- Node.js has a built-in module called HTTP, which allows Node.js to transfer data over the Hyper Text Transfer Protocol (HTTP)

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

- The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client
- Use the createServer() method to create an HTTP server:

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

//create a server object:
http.createServer(function (req, res) {
  res.write('Hello World!'); //write a response to the client
  res.end(); //end the response
}).listen(8080); //the server object listens on port 8080
```

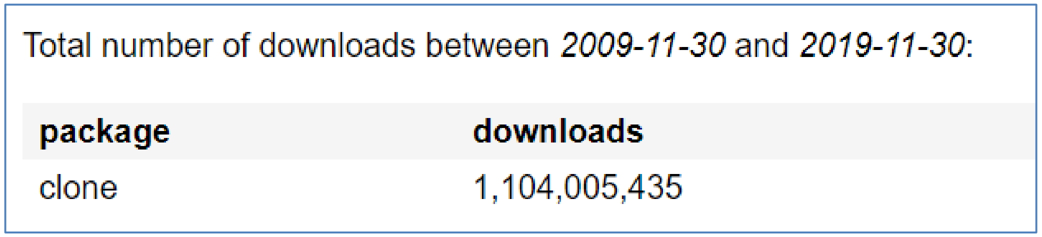
The function passed into the http.createServer() method, will be executed when someone tries to access the computer on port 8080.

https://www.w3schools.com/nodejs/nodejs_http.asp

URL Module

- The URL module splits up a web address into readable parts.
- Parse an address with the url.parse() method, and it will return a URL object with each part of the address as properties:

```
var url = require('url');
var adr = 'http://localhost:8080/default.htm?year=2017&month=february';
var q = url.parse(adr, true);

console.log(q.host); //returns 'localhost:8080'
console.log(q.pathname); //returns '/default.htm'
console.log(q.search); //returns '?year=2017&month=february'

var qdata = q.query; //returns an object: { year: 2017, month: 'february' }
console.log(qdata.month); //returns 'february'
```



Total number of downloads between *2009-11-30* and *2019-11-30*:

package	downloads
clone	1,104,005,435

Node.js NPM

- ❑ NPM is a package manager for Node.js packages (modules)
- ❑ www.npmjs.com hosts thousands of free packages to download and use.
- ❑ The NPM program is installed on your computer when you install Node.js
- ❑ Example:
  - Using a package that converts the output "Hello World!" into upper-case letters:

```
var http = require('http');
var uc = require('upper-case');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write(uc("Hello World!"));
  res.end();
}).listen(8080);
```

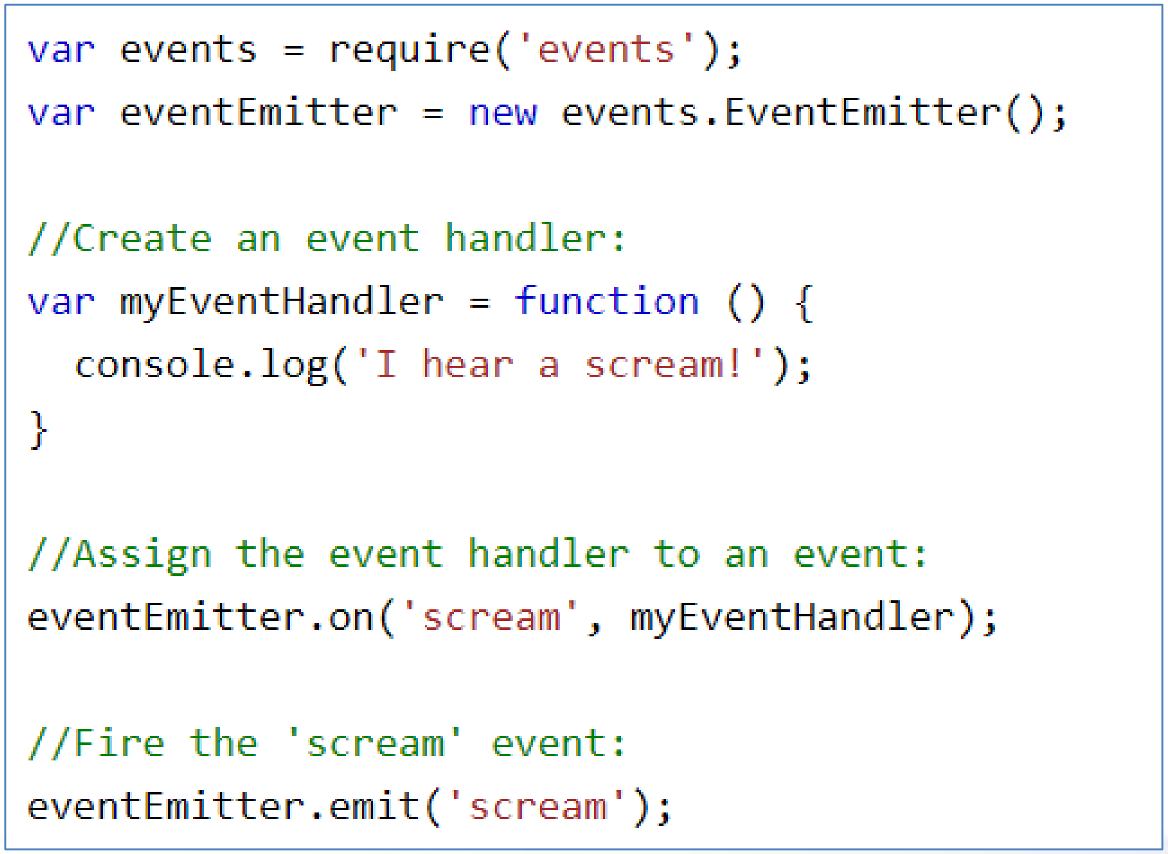
Events

- Node.js is perfect for event-driven applications.
- Example:
  - Objects in Node.js can fire events, like the *readStream* object fires events when opening and closing a file:

```
var fs = require('fs');
var rs = fs.createReadStream('./demofile.txt');
rs.on('open', function () {
  console.log('The file is open');
});
```

- You can assign event handlers to your own events with the EventEmitter object
- To fire an event, use the emit() method

EventEmitter object

- In the example below we have created a function that will be executed when a "scream" event is fired.
- 

```
var events = require('events');
var eventEmitter = new events.EventEmitter();

//Create an event handler:
var myEventHandler = function () {
  console.log('I hear a scream!');
}

//Assign the event handler to an event:
eventEmitter.on('scream', myEventHandler);

//Fire the 'scream' event:
eventEmitter.emit('scream');
```

Upload Files

- There is a good module for working with file uploads, called "Formidable"
- Steps:
  1. Create an Upload Form

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

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write('<form action="fileupload" method="post"
enctype="multipart/form-data">');
  res.write('<input type="file" name="filetoupload"><br>');
  res.write('<input type="submit">');
  res.write('</form>');
  return res.end();
}).listen(8080);
```

Upload Files...

2. Parse the Uploaded File

```
var http = require('http');
var formidable = require('formidable');

http.createServer(function (req, res) {
  if (req.url == '/fileupload') {
    var form = new formidable.IncomingForm();
    form.parse(req, function (err, fields, files) {
      res.write('File uploaded');
      res.end();
    });
  } else {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write('<form action="fileupload" method="post" enctype="multipart/form-data">');
    res.write('<input type="file" name="filetoupload"><br>');
    res.write('<input type="submit">');
    res.write('</form>');
    return res.end();
  }
}).listen(8080);
```

Upload Files…

3. Save the File

- When a file is successfully uploaded to the server, it is placed on a temporary folder
- The path to this directory can be found in the "files" object, passed as the third argument in the parse() method's callback function
- To move the file to the folder of your choice, use the File System module, and rename the file

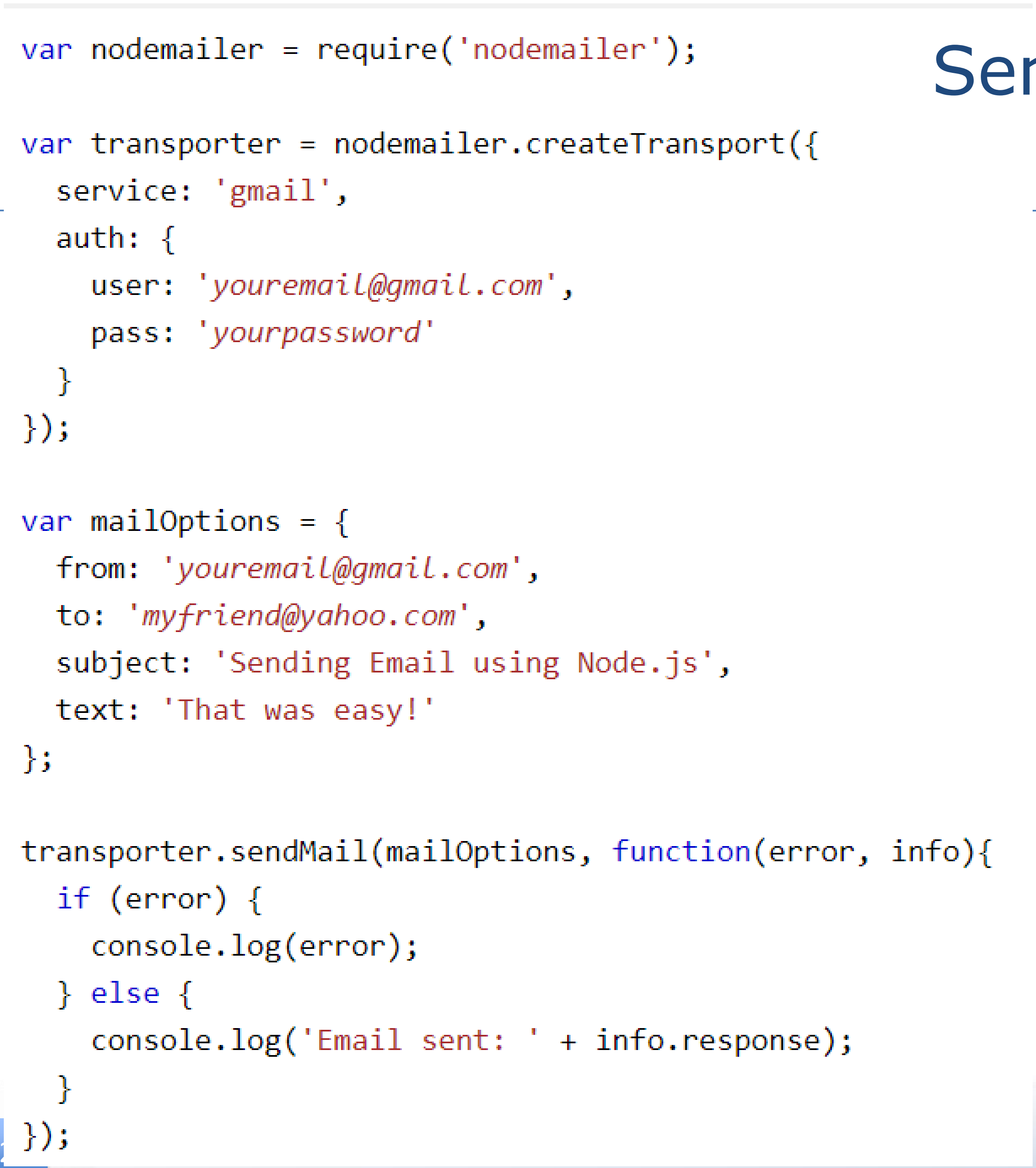
Upload Files… (3. Save the File)

```
var http = require('http');
var formidable = require('formidable');
var fs = require('fs');

http.createServer(function (req, res) {
  if (req.url == '/fileupload') {
    var form = new formidable.IncomingForm();
    form.parse(req, function (err, fields, files) {
      var oldpath = files.filetoupload.path;
      var newpath = 'C:/Users/Your Name/' + files.filetoupload.name;
      fs.rename(oldpath, newpath, function (err) {
        if (err) throw err;
        res.write('File uploaded and moved!');
        res.end();
      });
 });
  } else {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write('<form action="fileupload" method="post" enctype="multipart/form-data">');
    res.write('<input type="file" name="filetoupload"><br>');
    res.write('<input type="submit">');
    res.write('</form>');
    return res.end();
  }
}).listen(8080);
```

Send an Email

- The Nodemailer module makes it easy to send emails from your computer.
- The Nodemailer module can be downloaded and installed using npm
  - *C:\npm install nodemailer*
- After you have downloaded the Nodemailer module, you can include the module in any application:
- Use the username and password from your selected email provider to send an email.
- This lecture will show you how to use your Gmail account to send an email:

Send an Email…



```
var nodemailer = require('nodemailer');

var transporter = nodemailer.createTransport({
  service: 'gmail',
  auth: {
    user: 'youremail@gmail.com',
    pass: 'yourpassword'
  }
});

var mailOptions = {
  from: 'youremail@gmail.com',
  to: 'myfriend@yahoo.com',
  subject: 'Sending Email using Node.js',
  text: 'That was easy!'
};

transporter.sendMail(mailOptions, function(error, info){
  if (error) {
    console.log(error);
  } else {
    console.log('Email sent: ' + info.response);
  }
});
```

Send an Email…

- Multiple Receivers

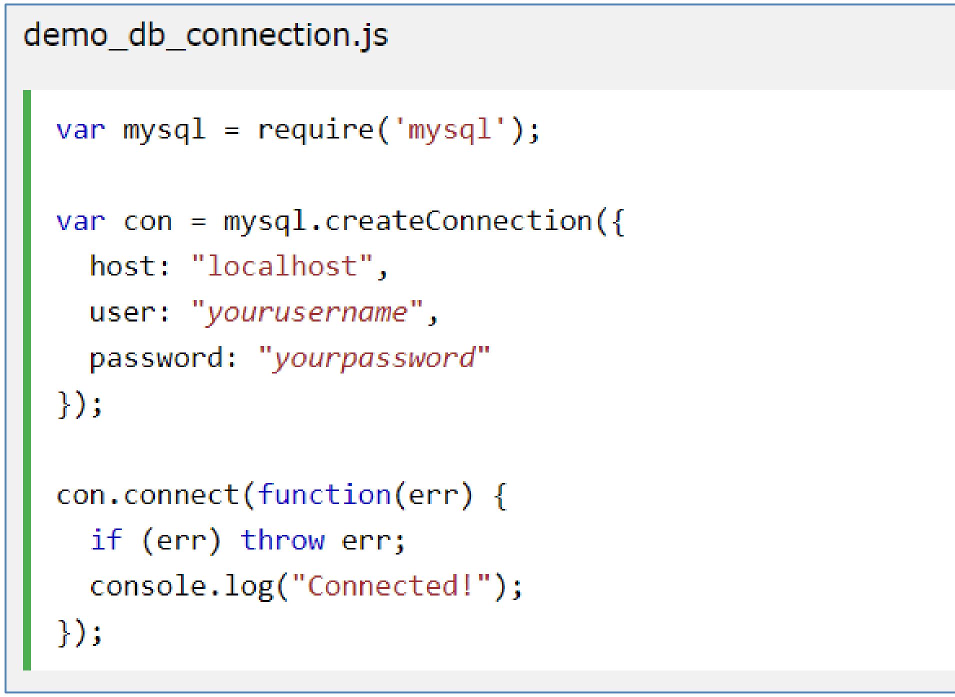
```
var mailOptions = {
  from: 'youremail@gmail.com',
  to: 'myfriend@yahoo.com, myotherfriend@yahoo.com',
  subject: 'Sending Email using Node.js',
  text: 'That was easy!'
}
```

- Send HTML

```
var mailOptions = {
  from: 'youremail@gmail.com',
  to: 'myfriend@yahoo.com',
  subject: 'Sending Email using Node.js',
  html: '<h1>Welcome</h1><p>That was easy!</p>'
}
```

MySQL

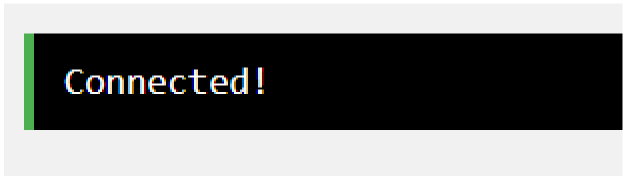
- ❑ Node.js can be used in database applications.
- ❑ To work with MySQL, you need to install it first in your machine
  - C:\npm install mysql
- ❑ Create Connection



demo_db_connection.js

```
var mysql = require('mysql');

var con = mysql.createConnection({
  host: "localhost",
  user: "yourusername",
  password: "yourpassword"
});

con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
});
```

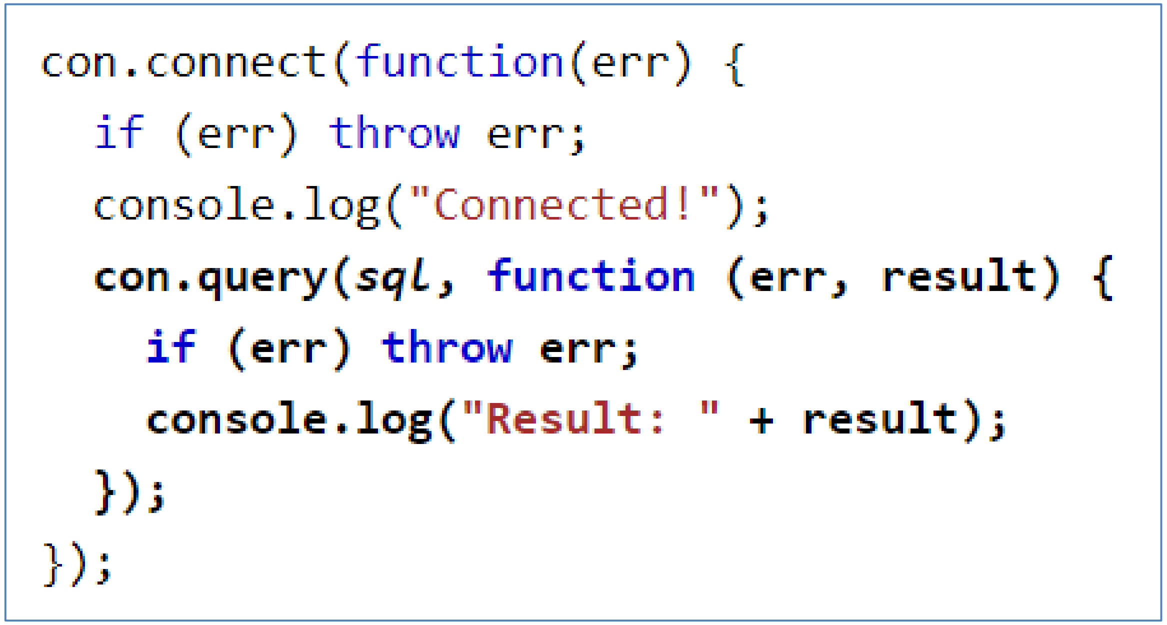
- ❑ Run "demo_db_connection.js"
  - C:\node demo_db_connection.js



```
Connected!
```

Query a Database

- The query method takes an sql statements as a parameter and returns the result.



```
con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("Result: " + result);
  });
});
```

Creating a Database

- ❑ Create a database named "mydb"

```
var mysql = require('mysql');

var con = mysql.createConnection({
  host: "localhost",
  user: "yourusername",
  password: "yourpassword"
});

con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
  con.query("CREATE DATABASE mydb", function (err, result) {
    if (err) throw err;
    console.log("Database created");
  });
});
```

MySQL Insert

- Insert a record in the "customers" table

```
var mysql = require('mysql');

var con = mysql.createConnection({
  host: "localhost",
  user: "yourusername",
  password: "yourpassword",
  database: "mydb"
});

con.connect(function(err) {
  if (err) throw err;
  console.log("Connected!");
  var sql = "INSERT INTO customers (name, address) VALUES ('Company Inc', 'Highway 37')";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("1 record inserted");
  });
});
```

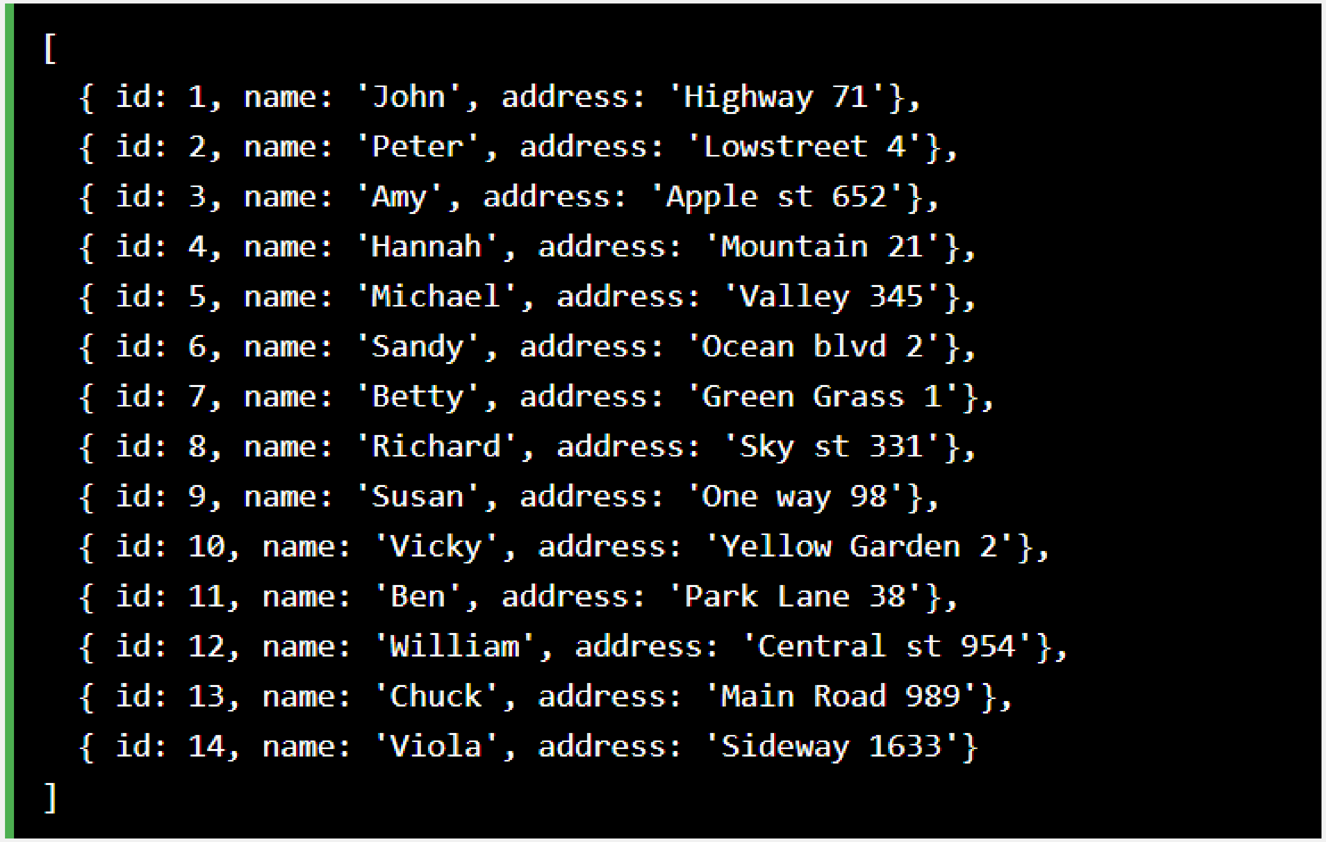
MySQL Select

- Select all records from the "customers" table, and display the result object

```
var mysql = require('mysql');

var con = mysql.createConnection({
  host: "localhost",
  user: "yourusername",
  password: "yourpassword",
  database: "mydb"
});

con.connect(function(err) {
  if (err) throw err;
  con.query("SELECT * FROM customers", function (err, result, fields) {
    if (err) throw err;
    console.log(result);
  });
});
```

MySQL Select ....

- Run "demo_db_select.js“
  - *C:\node demo_db_select.js*
- Which will give you this result:



```
[
  { id: 1, name: 'John', address: 'Highway 71'},
  { id: 2, name: 'Peter', address: 'Lowstreet 4'},
  { id: 3, name: 'Amy', address: 'Apple st 652'},
  { id: 4, name: 'Hannah', address: 'Mountain 21'},
  { id: 5, name: 'Michael', address: 'Valley 345'},
  { id: 6, name: 'Sandy', address: 'Ocean blvd 2'},
  { id: 7, name: 'Betty', address: 'Green Grass 1'},
  { id: 8, name: 'Richard', address: 'Sky st 331'},
  { id: 9, name: 'Susan', address: 'One way 98'},
  { id: 10, name: 'Vicky', address: 'Yellow Garden 2'},
  { id: 11, name: 'Ben', address: 'Park Lane 38'},
  { id: 12, name: 'William', address: 'Central st 954'},
  { id: 13, name: 'Chuck', address: 'Main Road 989'},
  { id: 14, name: 'Viola', address: 'Sideway 1633'}
]
```