# Department of Computing

**CS-213: Advanced Programming**

**Class: BSCS 7AB**

# Lab 4: Introduction to Node.js

**Date: 26rd September, 2019**

**Time: 10:00-01:00pm & 02:00-05:00pm**

# Instructor: Dr. Sidra Sultana

**Lab Engineer: Ms. Ayesha Asif**

# 

# Lab 4: Introduction to Node.js

**Introduction**

This lab is about the installation and introduction to node.js environment. Node.js is an open source server environment. It runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.). Node.js uses JavaScript on the server.

**Node.js uses asynchronous programming!**

A common task for a web server can be to open a file on the server and return the content to the client.

**Objectives**

This lab will get you familiar with the node.js environment.

**Tools/Software Requirement**

Node.js, Notepad

**Description**

**Helping Material**

Slides of Lecture 4 & 5

<https://www.w3schools.com/nodejs/nodejs_get_started.asp>

**Lab Tasks**

**Task 1:** Download Node.js from the official Node.js web site: [https://nodejs.org](https://nodejs.org/).

**Task 2:** Once you have downloaded and installed Node.js on your computer, let's try to display "Hello World" in a web browser.

Hint:

Create a Node.js file named "myfirst.js", and add the code.

Save the file on your computer: C:\Users\Your Name\myfirst.js

The file you have just created must be initiated by Node.js before any action can take place.

Start your node.js command line interface, write node myfirst.js and hit enter

**Task 3:** Create a module that returns the current date and time. Save the code in a file called "myfirstmodule.js".

**Task 4:** Use the module "myfirstmodule" of date and time in a Node.js file.

**Node.js as a Web Server**

**Task 5:** 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.

**Task 6:** Add an HTTP Header

Hint: If the response from the HTTP server is supposed to be displayed as HTML, you should include an HTTP header with the correct content type like:

**res.writeHead(200, {'Content-Type': 'text/html'});**

The first argument of the res.writeHead() method is the status code, 200 means that all is OK, the second argument is an object containing the response headers.

**Node.js as a File Server**

The Node.js file system module allows you to work with the file system on your computer. To include the File System module, use the require() method:

var fs = require('fs');

Common use for the File System module:

* Read files
* Create files
* Update files
* Delete files
* Rename files

**Task 7:** Create a Node.js file that reads the HTML file, and return the content.

**Task 8:** Create a new file using

* appendFile() method
* open() method
* writeFile() method

**Task 9:** Append "This is my text." to the end of the file "mynewfile1.txt".

**Task 10:** Replace the content of the file "mynewfile3.txt".

**Task11:** Delete "mynewfile2.txt".

**Task 12:** Rename "mynewfile1.txt" to "myrenamedfile.txt".

|  |
| --- |
| Solution |
| Task 1,2,3,4,5,6 Code:  Module code:  exports.myDateTime = function () {  return Date();  };  Nodejs file code:  var http=require('http');  var date = require('./myfirstmodule');  http.createServer(function (req,res){  res.writeHead(200,{'Content-Type': 'text/html'});  res.write("The date and time is: "+ date.myDateTime());  res.end('Hello world!');  }).listen(8080);  Task 1,2,3,4,5,6 Output Screenshot:    Task 8 Code:   1. APPENDFILE METHOD:   var fs=require ('fs');  fs.appendFile('mynewfile1.txt','file created using appendfile!',  function(err){  if(err) throw err;  console.log('Created!');  });   1. OPEN METHOD:   var fs=require ('fs');  fs.open('mynewfile2.txt','w',  function(err){  if(err) throw err;  console.log('File Created using open method!');  });   1. WRITEFILE METHOD:   var fs=require ('fs');  fs.writeFile('mynewfile2.txt','w',  function(err){  if(err) throw err;  console.log('File Created using write method!');  });  Task 8 Output Screenshot:        Task 9 Code:  var fs=require ('fs');  fs.appendFile('mynewfile1.txt','This is my text',  function(err){  if(err) throw err;  console.log('this is my text added to file 1!');  });  Task 9 Output Screenshot:      Task 10 code:  var fs = require('fs');  fs.writeFile('mynewfile3.txt', 'Replaced text!', function (err) {  if (err) throw err;  console.log('replaced!');  });  Task 10 code output:    Task 11 code:  var fs = require('fs');  fs.unlink('mynewfile2.txt', function (err) {  if (err) throw err;  console.log('File deleted!');  });  Task 11 code output:    Task 12 code:  var fs = require('fs');  fs.rename('mynewfile1.txt', 'myrenamedfile.txt', function (err) {  if (err) throw err;  console.log('File Renamed!');  });  Task 12 code output: |

### Deliverables

Compile a single word document by filling in the solution part and submit this Word file on LMS. This lab grading policy is as follows: The lab is graded between 0 to 10 marks. The submitted solution can get a maximum of 5 marks. At the end of each lab or in the next lab, there will be a viva related to the tasks. The viva has a weightage of 5 marks. Insert the solution/answer in this document. You must show the implementation of the tasks in the designing tool, along with your complete Word document to get your work graded. You must also submit this Word document on the LMS. In case of any problems with submissions on LMS, submit your Lab assignments by emailing it to Ms. Ayesha Asif: [ayesha.asif@seecs.edu.pk](mailto:ayesha.asif@seecs.edu.pk).