Exercise #1:

What is “server side” processing mean? How can this make web pages dynamic?

You get it from the server, servers are reliable, secure, redundant.

Exercise #2:

Create a Lab4 folder in your repo and a new file Ex2.js and copy the following into this file:

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

*//create a server object:*

http.createServer(**function** (req, res) {

console.log(req.headers); *//output the request headers to the console*

res.writeHead(200, { 'Content-Type': 'text/html' }); *// set MIME type to HTML*

res.write('<h1>The date is: ' **+** Date.now() **+** '</h1>'); *//write a response to the client*

res.end(); *//end the response*

}).listen(8080); *//the server object listens on port 8080*

console.log('Hello world HTTP server listening on localhost port 8080');

Open a terminal and run it using node.js. OPen a browser and make a request to localhost:8080

1. Why is this a “dynamic” web page?

Bc it changes over time

b) Does the request matter? Where is the route handled?

The route is sent to the webserver to update time

c) When you do a “view source” in the browser, where is the Javascript code?

It turned into a number bc it came from a server

Exercise #3:

Make a copy of the file Ex2.js and change the name to Ex3.js then change line:

res.write('<h1>The date is: ' **+** Date.now() **+** '</h1>'); *//write a response to the client*

to

res.write('<h1>The date is:<script>document.write( Date.now() );</script></h1>'); *//write a response to the client*

Save the file, stop node.js and re-run (the new code needs to be loaded into Node.js)

Now, explain why this is the same result as the previous dynamic web page example, but also explain how it is different.

**Before we were just loading a string then we changed it to load a program**

Exercise #4: Page redirection

It is is common that a web app will need to direct the user to a different page other than the one requested. Say you want the user to be sent to google.com after requesting something from the server. This can be done in various different ways as we will explore. First make a copy of the file Ex3.js and change the name to Ex4.js.

Task1: Redirect on the client with HTML.

Change the string inside the res.write() to respond to HTTP requests with

<META http- equiv ="refresh" content="0;URL=http://www.google.com">

Stop and restart node.js and go to localhost:8080 to verify you get directed to google.com. Now stop node.js and go to localhost:8080 and verify you get a page not found.

Task 2: Redirect on the client with Javascript:

Copy the res.write() line and paste it below the original. In the copied change the string inside the res.write() of the copied line to respond to HTTP requests with

window.location = "http://www.apple.com";

Stop and restart node.js and go to localhost:8080 to verifiy you go to apple.com. Notice that you no longer go to google.com even though the HTML redirect is still there! Now stop node.js and go to localhost:8080 and verify you go get a page not found.

Task 3: Redirect by changing the routing on the server:

Copy the res.writeHead() line and replace the string in the copy with:

res.writeHead(301, { "Location": "http://amazon.com"});

Stop and restart node.js and go to localhost:8080. Notice that you no longer go to apple.com even though the Javascript redirect is still there. Now stop node.js and try going to localhost:8080. Did you get a page not found?

Questions to discuss: Explain how these each work differently and when you might choose one way over another. Which ones would enabel you to use the broswer “back” or history? Hint: the server side redirect can not be stopped or changed by the user.

Exercise #5: Mixing client and server processing to modify DOM objects

First make a copy of the file Ex3.js (note, this is from **Excercise 3** not 4) and change the name to Ex5.js. In your server code, replace

res.write('<h1>The date is:<script>document.write( Date.now() );</script></h1>'); *//write a response to the client*

with

res.writeHead(200, { 'Content-Type': 'text/html' }); *// set MIME type to HTML*

res.write("<body></body><script>document.body.style.backgroundColor= 'rgb("

**+** Math.floor(Math.random() **\*** 255) **+** ','

**+** Math.floor(Math.random() **\*** 255) **+** ','

**+** Math.floor(Math.random() **\*** 255) **+** ")';</script>"

); *//write a response to the client*

Stop node.js and restart with the new code. View the page then try refreshing a few time. Now view the page source. Explain how this blends both client and server side processing.

**Server is serving up the script and the client is running the script loading a script on the client side and produces a different result so you should never see the same server twice.**