***Documentation Packet [ 03 06 09 24 ] Sep 6th 2024***

|  |  |
| --- | --- |
| Student Name: |  |
| Goals:  1. Create a basic NodeJS HTTP server with ExpressJS 2. Use Gamepad controls in a webpage | Events:  1. DocPac Due *Next Monday* |
| Included Documentation  1. Button Masher 2. NEE Stack Intro | Required Documentation:  1. Button Masher 2. NEE Stack Intro |
| Changes/Notes:  * None this week | |

# NEE Stack Intro

Create a new NodeJS project to practice NodeJS, ExpressJS, and EJS. Use your senior mentor and online resources to complete the following tasks. As a matter of self-educating, practice reading online tutorials and documentation instead of asking ChatGPT.

1. Create and initialize a NodeJS project
2. Install ExpressJS into your project
3. Import ExpressJS module
4. Configure ExpressJS to use the EJS templating engine
5. Create a HTTP Listen Server with Express
6. Create a root (‘/’) GET endpoint that responds by **sending** a premade HTML homepage
7. Create another GET endpoint that handles the request in the following way:
   1. Create a “name” variable in the endpoint’s callback function whose value is “Guest”
   2. If the request had a “name” **query parameter**, change “name” to the value of that parameter
   3. Respond by rendering an EJS template of a basic web page
      1. The webpage must have a header that says “Hello, <name>”, where <name> is the value of the “name” variable.
      2. Test to make sure your name appears in the template if you give the endpoint a “name” query parameter ( *localhost:3000/endpoint?name=yourname* )

When you are finished:

1. Delete the ‘node\_modules’ folder
2. Rename your project folder to “FirstnameLastname”, and move it to this DocPac’s “neepractice” folder. If this folder does not exist, create it exactly as worded here.
3. Commit your changes to your DocPac repo fork. Open a Pull Request to have your work merged into the class’s DocPac Repo (ask a senior if you need assistance)

# Button Masher

Create a web page game that can test how fast you can mash the “A” button of an Xbox controller. The catch is you only have 20 seconds, and you have to pushing the Joystick in the right direction

Use this site for more details on using Gamepads in Javascript browser applications:

<https://www.javascripture.com/Gamepad>

1. In an HTML file, create a simple web page
   1. Put a div layer in the body of the web page that has an id.
      1. This guide will call this box the “score box”
   2. Put another div layer in the body of the web page that has an id.
      1. This guide will call this box the “timer box”
   3. Put another div layer in the body of the web page that has an id.
      1. This guide will call this box the “direction box”
2. In the <script> tag:
   1. Create a variable called “score”, a variable called “timer” and a variable called “direction”.
      1. Score and direction start at 0
      2. Timer starts at 20
   2. Using a setInterval() function, create a function that decreases the timer variable by one every second if it’s greater than 0. It will then change the innerHTML property of the timer box by its current value minus 1
   3. Using a setInterval() function, create a function that changes the direction to a random integer between 0 and 3 every 2 seconds. It will then change the innerHTML property of the direction box to the name of the direction listed below
      1. 0 is up
      2. 1 is right
      3. 2 is down
      4. 3 is left
3. Set up your script to detect gamepad button presses
   1. Each time the “A” button is pressed:
      1. If the time is greater than 0, and the direction of the left joystick matches the direction, increase the score by 1, then change the innerHTML property of the score box by its current value plus 1
   2. If the “Start” button is pressed, refresh the web page in the browser

When you are finished:

1. Delete the ‘node\_modules’ folder
2. Rename your project folder to “FirstnameLastname”, and move it to this DocPac’s “buttonmasher” folder. If this folder does not exist, create it exactly as worded here.
3. Commit your changes to your DocPac repo fork. Open a Pull Request to have your work merged into the class’s DocPac Repo (ask a senior if you need assistance)

# Reflection

**What did you do to understand the instructions in your assignments this week? What were the advantages/disadvantages of doing it this way?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**How are you doing on managing and completing all of your class’s work this year? What are the advantages/disadvantages of your current strategy to complete the work?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**What is one project you are actively working on in programming outside of the classwork? What strategy do you have to complete it? If no project, why not?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**Was the DocPac better or worse for you this week? What made it different for you?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

# A picture containing text, monitor, screen, clipart Description automatically generatedGrading

|  |  |  |
| --- | --- | --- |
| 10 | You went above and beyond expectations. You applied knowledge that was not taught in this class in addition to what was taught. Additional rewards are given | * ***If the assignment does not have its own rubric, it will default to the rubric on the left.*** * All assignments start at 10/10 possible points * 1 point is deducted per infraction   + Lateness   + Mistakes   + Unprofessionalism   + Not following instructions * Outstanding submissions, or submissions on assignments not marked in “Required Documentation” can reward pogs |
| 10 | You performed as well as can be expected for this class. You show a complete understanding and made no mistakes. You have mastered the subject. |
| 8 | Assignment is complete. You show a good understanding of the subject, but there are mistakes or minor incorrect details. You are ready to move to new subjects. |
| 7 | You show and understanding of the subject, but there are serious errors, or there are pieces you can practically use without understanding them. Remediation needed. |
| 6 | Assignment is incomplete but/or you showed that you understand at least the fundamentals of the subject. Assignment is low effort. Serious need of remediation. |
| 5 | You show minimum effort, assignment is incomplete, or have serious mistakes. You did not demonstrate that you understand the content or purpose of the submission. |
| 0 | The work was not submitted, damaged, seriously incorrect, or unprofessional. The submission is rejected. |
|  |  |  |

# Button Masher

|  |  |  |  |
| --- | --- | --- | --- |
| Correct file location and name | Pull Request Accepted | Can read direction/button input and increase score | Timer ending prevents score from increasing |

# NEE Stack Intro

|  |  |  |  |
| --- | --- | --- | --- |
| Pull Request Accepted  (node\_modules deleted) | Server can run | Can send an html file when you go to “/” | Second endpoint can read the user’s name with a query parameter |

# Reflection

|  |  |  |  |
| --- | --- | --- | --- |
| Selected an answer for each question that is unique to you and this week | Answered every question in each prompt | Answers were not repeats of previous weeks | Answers were not copies of assigned work this week |