Task 2: Building a Simple HTTP Server

Objective:

Build a basic HTTP server using Node.js that responds with "Hello, World!". Ensure the server runs without errors and can be accessed via a web browser.

Prerequisites:

- Basic understanding of JavaScript and Node.js
- Node.js installation

Concepts:

Creating an HTTP Server:

Node.js provides the `http` module to create an HTTP server. The server listens for incoming requests and sends responses.

Example:

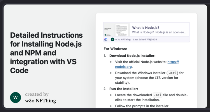
```
// httpServerExample.js
const http = require('http');

// Create an HTTP server
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello, World!\n');
});

// Define the port to listen on
const port = 3000;

// Start the server and listen on the specified port
server.listen(port, () => {
  console.log(`Server running at http://localhost:${port}/`);
});
```

Setup:

1. **Install Node.js:**
   Ensure Node.js is installed on your machine. You can access the instructions here:
   - 
     Detailed Instructions for Installing Node.js and NPM and integration with VS Code

     Detailed Instructions for Installing Node.js and NPM and integration with...
     For Windows: — Download Node.js Installer: — Visit the official Node.js website: http...
     w3o NFThing Last Edited 7/3/2024

Tasks:

HTTP Server:

1. **Task:**
   - Create a file named `httpServer.js`.
   - Write code to:
     - Import the `http` module.
     - Create an HTTP server that responds with "Hello, World!" for every request.
     - Listen on port 3000.
2. **Outcome:**
   - Ensure the server responds with "Hello, World!" when accessed via a browser.

Instructions:

1. **Perform the following tasks:**
   - Write the required code in a file named `httpServer.js`.
   - Run the file using Node.js to ensure the server starts without errors and demonstrates the use of Node.js HTTP server features.
2. **Running the server:**
   - Use the following command to run your server:

     ```
     node httpServer.js
     ```

   - Open a web browser and navigate to `http://localhost:3000`.
   - You should see the message "Hello, World!" displayed in the browser.

Resources:

- HTTP | Node.js v22.4.0 Documentation
  This module, containing both a client and server, can be imported via — require('node:http') (Commo...
  https://nodejs.org/api/http.html
- Index | Node.js v22.3.0 Documentation
  https://nodejs.org/en/docs/

Videos:

Web Dev Simplified 8:57min 543,112 Views 10,623 Likes

3. **GitHub Instructions:**
   - **Open in Visual Studio Code:**
     - After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, Visual Studio Code (VSCode) will open the repository directly.
     - If prompted, select "Open" or "Allow" to open the repository in VSCode.
   - **Complete the Task:**
     - Write your solution in `httpServer.js` .
   - **Run and Test Your Code:**
     - Run your code to ensure it works correctly using:

       ```
       node httpServer.js
       ```
   - **Commit Your Changes:**
     - Commit your changes with a meaningful message:

       ```
       git commit -m "Completed Simple HTTP Server task"
       ```
   - **Push Your Changes to Your Forked Repository:**
     - Push your changes to your forked repository:

       ```
       git push origin main
       ```

- **Create a Pull Request:**
    - Go to your forked repository on GitHub.
    - Click on the "Pull Requests" tab.
    - Click the "New Pull Request" button.
    - Ensure the base repository is the original template repository and the base branch is `main`.
    - Ensure the head repository is your forked repository and the compare branch is `main`.
    - Click "Create Pull Request".
    - Add a title and description for your pull request and submit it.

Summary of Commands:

```
# Fork the repository on GitHub

# Clone the forked repository
git clone https://github.com/your-github-username/repository-name.git
cd repository-name

# Complete the task by writing and running the code in the specified files

# Add, commit, and push your changes
git commit -m "Completed Simple HTTP Server task"
git push origin main

# Create a pull request on GitHub
```