Method 1: Communicating Using Web Sockets

Web Socket is a protocol which provides a full duplex(multiway) communication that is it allows communication in both directions simultaneously. **ws** is a simple to use, blazing fast, and thoroughly tested WebSocket client and server implementation.

There are 3 basic ws functions:

- ws.onopen : emmited when connected
- ws.send : sending a send event to websocket server
- ws.onmessage : event emmited when receiving message

Chalk is a library that provides a simple and easy to use interface for applying ANSI colours and styles to your command-line output. The Node.js module to help us accomplish custom formatting of messages

Installing

1. WebSocket module

   npm install ws

2. Chalk module

   npm install chalk

API Docs

1. /doc/ws.md for Node.js-like documentation of ws classes and utility functions.
2. /chalk For Chalk Module and its specifications.

Run Code

1. Open 2 Terminals
2. Navigate to the folder where the demo_server.js and demo_client.js files are saved.
3. On one terminal run the following command (run Server file first):

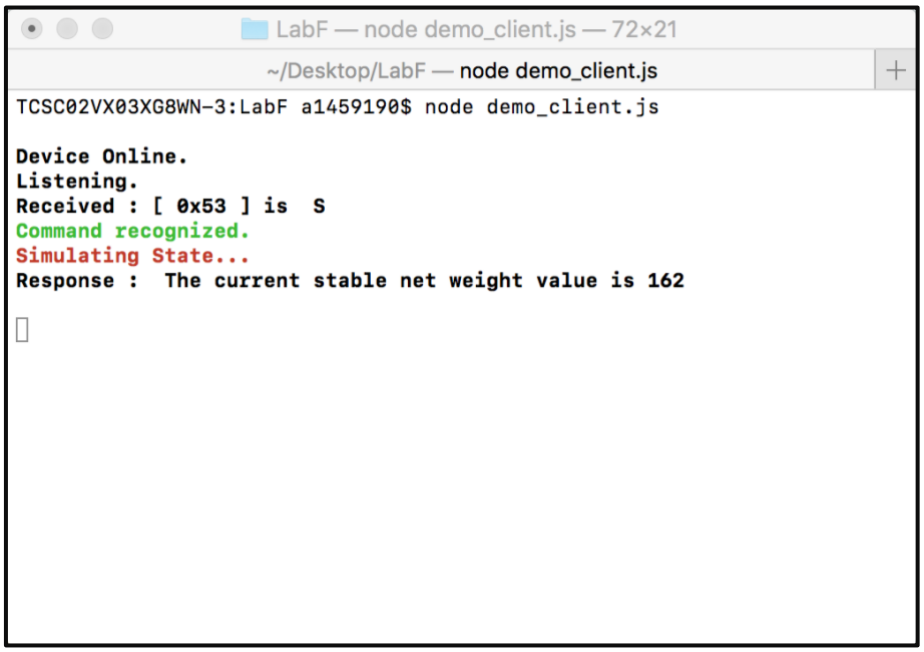
node demo_server.js

4. On second terminal run the client file using the following command:

   node demo_client.js

5. Terminal Output for the Server:

6. Terminal Output for the Client:



Method 2: Communicating Using Node-IPC

Node-IPC a nodejs module for local and remote Inter Process Communication with full support for Linux, Mac and Windows. It also supports all forms of socket communication from low level unix and windows sockets to UDP and secure TLS and TCP sockets.

Chalk is a library that provides a simple and easy to use interface for applying ANSI colours and styles to your command-line output. The Node.js module to help us accomplish custom formatting of messages

Installing

3. WebSocket module

   npm install node-ipc

4. Chalk module

   npm install chalk

API Docs

3. /node-ipc for Node.js-like documentation of ws classes and utility functions.
4. /chalk For Chalk Module and its specifications.

Run Code

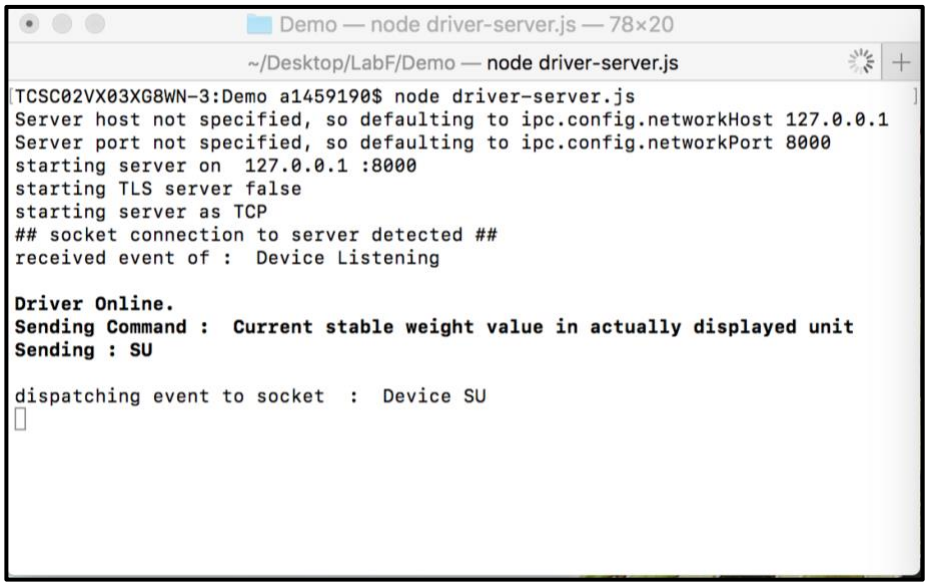
7. Open 2 Terminals
8. Navigate to the folder where the driver-server.js and device-client.js files are saved.
9. On one terminal run the following command (run Server file first):

   node driver-server.js

10. On second terminal run the client file using the following command:

    node device-client.js

11. Terminal Output for the Server:



12. Terminal Output for the Client: