Full Stack Web Framework with BBG

*** This guide will be for mac/linux (All commands will be UNIX). Try Windows at your own risk.

Intro to Meteor

Throughout the course of the semester you will most likely need to use some sort of website to interact with your Beaglebone. This guide will show you the simplest way to achieve this functionality.

Why Meteor

Meteor is a full stack framework that requires the use of only one language, javascript. For this reason, it's really easy to pick up for beginners. Meteor is also reactive which means that whenever there's a change in the server it's reflected in the client right away.

Installation

Mac

1. Run the command:

```
> curl https://install.meteor.com/ | sh
```

That's it! You now have meteor.

Creating Your First Application

1. Open your the desired folder you want to create your app in
2. Run the command:

```
> meteor create test-app
```

3. You now have a meteor application. To see how it looks let's run it in localhost with the command

```
> cd test-app

> meteor
```

4. This will start you server at `localhost:3000` by default. To change it to your own custom port run the command

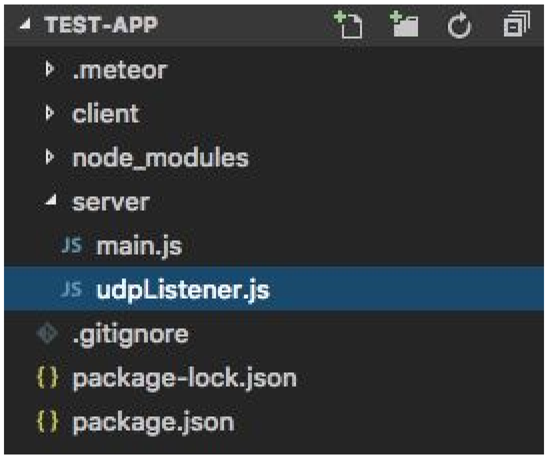
```
>  meteor run --port <your_port>
```

Here is an example to start your website at port 8080

```
> meteor run --port 8080
```

Communicate With BBG over UDP

Sending Data to Meteor Server


TEST-APP
.meteor
client
node_modules
server
main.js
udpListener.js
.gitignore
package-lock.json
package.json

1. Assuming that you are starting in the directory of the test-app we created in the previous step, go ahead and change directories to the server and make a new file called `udpListener.js`

```
> cd server

> mkdir udpListener.js
```

2. In `udpListener.js` we need to define an ip address, a port, and the protocol we want to use. This can be achieved by placing the following code into your udpListener

```
Meteor.startup(() => {

        var PORT = 12345

        var HOST = '127.0.0.1';

        var dgram = require('dgram');

        var server = dgram.createSocket('udp4');


        server.on('listening', function () {

                        var address = server.address();

                        console.log('UDP Server listening on ' + address.address + ":" +

                        address.port);

         });


        server.on('message', function (message, remote) {

                        console.log(remote.address + ':' + remote.port +' - ' + message);

        });

        server.bind(PORT, HOST);

});
```

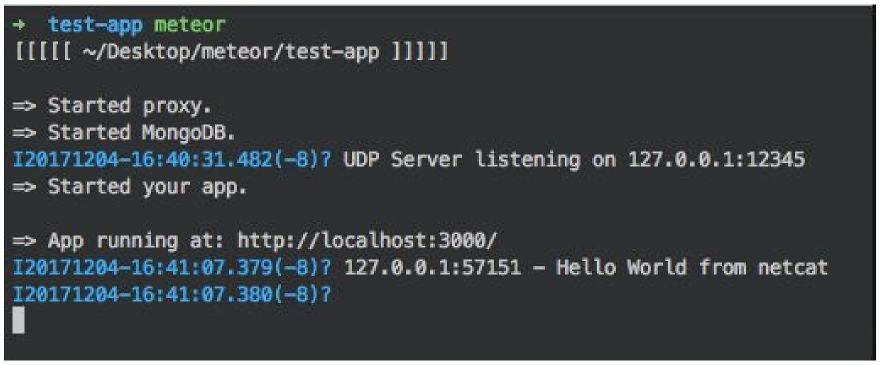
Notice that the server will be listening on port 12345 at ip 127.0.0.1

3. Test to see if this works with netcat

```
> nc -u 127.0.0.1 12345

> Hello World from netcat
```

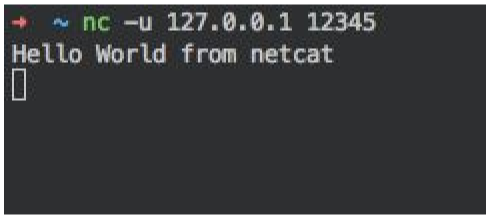
Here is what it should look like on netcat and on the meteor server terminal:

Meteor Server:



```
→  test-app meteor
[[[[[ ~/Desktop/meteor/test-app ]]]]]

=> Started proxy.
=> Started MongoDB.
I20171204-16:40:31.482(-8)? UDP Server listening on 127.0.0.1:12345
=> Started your app.

=> App running at: http://localhost:3000/
I20171204-16:41:07.379(-8)? 127.0.0.1:57151 - Hello World from netcat
I20171204-16:41:07.380(-8)?
```

Netcat



```
→  ~ nc -u 127.0.0.1 12345
Hello World from netcat
```

Receiving Data From Meteor Server

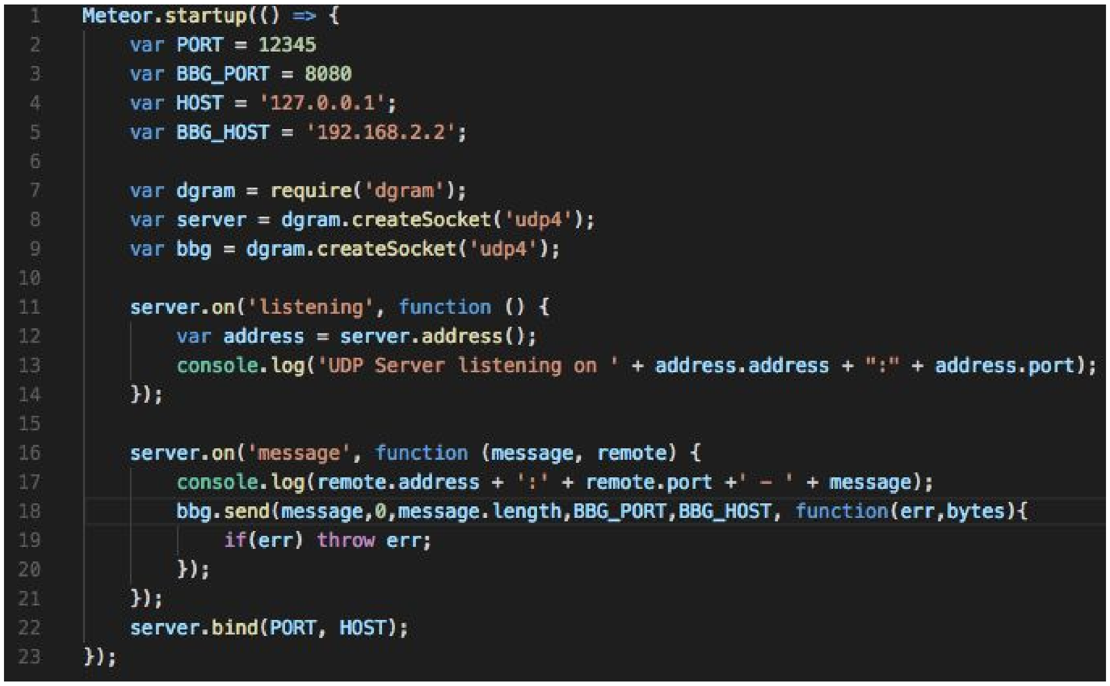
1. To make your meteor server send data you will need to define a new HOST and PORT. This will be the HOST and PORT your beaglebone will use so make sure it matches your beaglebone's C code. In my case I will use 192.168.2.2 at port 8080. You will also need to create a new socket. Just add these to the top of the meteor.create method

```
var BBG_PORT = 8080

var BBG_HOST = '192.168.2.2';

var bbg = dgram.createSocket('udp4');
```

2. Now, to send data you will need to use the method

```
bbg.send(message,0,message.length,BBG_PORT,BBG_HOST, function(err,bytes){
    if(err) throw err;
 });
```

Your file should now look like:



```
Meteor.startup(() => {
    var PORT = 12345
    var BBG_PORT = 8080
    var HOST = '127.0.0.1';
    var BBG_HOST = '192.168.2.2';

    var dgram = require('dgram');
    var server = dgram.createSocket('udp4');
    var bbg = dgram.createSocket('udp4');

    server.on('listening', function () {
        var address = server.address();
        console.log('UDP Server listening on ' + address.address + ":" + address.port);
    });

    server.on('message', function (message, remote) {
        console.log(remote.address + ':' + remote.port +' - ' + message);
        bbg.send(message,0,message.length,BBG_PORT,BBG_HOST, function(err,bytes){
            if(err) throw err;
        });
    });
    server.bind(PORT, HOST);
});
```

Troubleshooting

1. If you are having trouble installing meteor, refer to the official documentation here: https://www.meteor.com/install
2. If you are having issues starting the udp server make sure that you have the proper ip address and port. Run `> ifconfig` to see your ip.
3. If your beaglebone is not receiving messages from the meteor server then make sure that you have set up the correct host and port number on both the beaglebone c program and the `udpListener.js`
4. If your meteor project won't start you might have to delete the node_modules folder and install them again.

```
> rm -rf node_modules/ && npm install && meteor reset
```