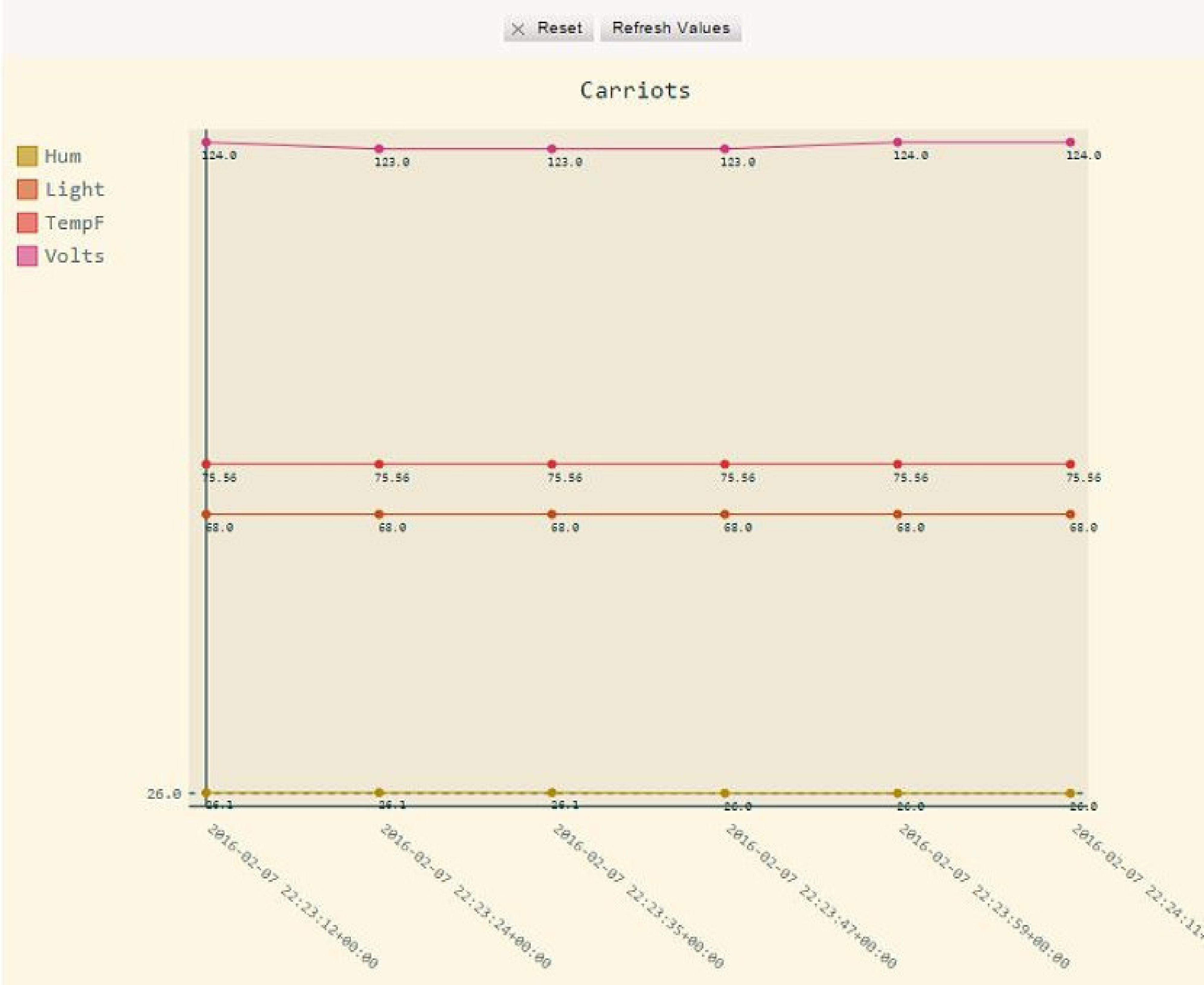
Reset
Refresh Values
Carriots
Hum
Light
TempF
Volts
124.0
123.0
123.0
123.0
124.0
124.0
75.56
75.56
75.56
75.56
75.56
75.56
68.0
68.0
68.0
68.0
68.0
68.0
26.0
26.1
26.1
26.1
26.0
26.0
26.0
2016-02-07 22:23:12+00:00
2016-02-07 22:23:24+00:00
2016-02-07 22:23:35+00:00
2016-02-07 22:23:47+00:00
2016-02-07 22:23:59+00:00
2016-02-07 22:24:11

```
/*
************************************
*  Carriots Cloud Station          *
*  ======================          *
*   By Roy H Guerra Jr.            *
*   2/7/16                         *
************************************
This code uses a Dagino Wi-Fi Board in "bridge mode" connected to an Arduino Uno to
monitor the light intensity, temperature, line Voltage and humidity.
The results are transferred to a cloud server at the Carriots Web Site and
is based on 1 of 4 Carriots protocalls.  In addition the code will be optomized to a
graph that was contructed with a Carriots graphing function.

Hardware Required:
------------------
- Arduino Uno R3
- Dragino Yun Wi-Fi Shield
- DHT22 temperature sensor
- Photocell
- Misc (project case, power supply, etc.)

 The circuit has the following features:
 1) Connects and Transmits / Recieves through the house wireless router
 2) Connects to a Carriots cloud server webpage that could be pulled up anywhere
 3) Sends a Text and / or E-Mail to your phone if the temperature is above 90 degrees F
 4) Uses an unregulated DC input through a volatge divider that is calibrated and scaled
```

```
to read VAC rms

 Note(s)-
 * See my seperate instructions on the Dragino setup and Arduino Yun library installation
 * Photocell tied to (+) and a series 10K resistor with one end tied to Gnd, other end to
"A0"
 * Line volts (DC) tied to "A1" (if this function is used, but also must have a battery
backup and voltage divider)
 * Set up the Carriots website (see Arduino tutorials)
 * Set debug to "true" to to read sensors and Carriots server messages for debugging
 * Once website and program are verified to run correctly, set debug to "false", and
reprogram if you want to.
 * When setting up carriots SMS on the WEB use the logic expression "context.data.TempF >
90", and
   use "001+phonenumber (no spaces).  For example, my phone number would be
"0014803092305"

 DHT 22 Sensor Wiring
 * Pin 1 (VDD)  = + 5v
 * Pin 2 = N/C
 * Pin 3 = Digital pin #7
 * Pin 4 = Gnd
*/

// Declare Libraries
// ------------------
#include <Bridge.h>
```

```
#include <Console.h>
#include <DHT.h>
#include <Process.h>
#include <YunClient.h>

// DHT sensor Characteristics
//-------------------------
#define DHTPIN 7     // Pin we are connected to
#define DHTTYPE DHT22   // DHT 22
DHT dht(DHTPIN, DHTTYPE);

// Light Level Sensor Characteristics
//--------------------------------
#define LIGHT_SENSOR_PIN A0 // Pin we are connected to

// Voltage Input
//--------------------------------
#define Voltage_PIN A1 // Pin we are connected to, Analog Input #1

// Define Carriots Information
//--------------------------
#define APIKEY "[redacted]"  //
TO BE REPLACED with your Carriots APIKEY
#define DEVICE "[redacted]"  // TO BE REPLACED with your Device's
ID developer

// Declare Variables
// =================
int LightLevel; // Light Intensity storage Variable
int VoltLevel; // Light Intensity storage Variable
float Light; // Light Intensity storage Variable after scaling
float Volts; // Ac Line Voltage variable
float Hum; // Humidity storage Variable
float TemperatureC;  // Temperature storage variable for Deg C
float TempF;  // Temperature storage variable for Deg F
String dataString = "";  //Constant to store the payload that will be sent

// Choose Debug Mode for trouble shooting?
//=========================================
boolean debug_mode = true; // Choose "true" for degugging

// Create a YunClient instance to communicate using the Yun's brighe & Linux OS.
YunClient client;

// Start of Main Program
// =====================
void setup() {
   dht.begin(); // Initialize DHT sensor
   Bridge.begin(); // Start Bridge
   Console.begin(); // Start Console (window used for troubleshooting once shield is
installed)
    if (debug_mode == true){  // Print if debug mode is true (on)
   Console.println("Setup complete. Waiting for sensor input...\n");
   }
```

```
  delay(1000); // 1 second delay
}

void loop() {

  SensorData(); // go to Sensor Function

  if (debug_mode == true){ // Print
   Console.println(F("Sending Stream"));
  }
   updateData(); // go to update data function
   sendData(); // go to send data function
  delay(120000); // Repeat every 2 minutes
}

void SensorData() {  //Read Sensor Function
  Hum = dht.readHumidity();  // Measure the humidity
  TemperatureC = dht.readTemperature(); // Measure the temperature
  TempF = ((TemperatureC * 9/5) + 32); // Convert temperature to degrees Fahrenheit
  LightLevel = analogRead(LIGHT_SENSOR_PIN); // Measure light level
  Light = map(LightLevel, 0, 1023, 0, 100);  // Scale light level (0-100%)
  VoltLevel = analogRead(Voltage_PIN); // Measure voltage level
  Volts = map(VoltLevel, 0, 1023, 0, 150);  // Scale voltage level (0-150)
  if (debug_mode == true){ // Print measurements
   Console.println("Humidity: ");
   Console.println(Hum);
   Console.println("Light level: ");
```

```
    Console.println(Light);
    Console.println("Temperature: ");
    Console.println(TempF);
    Console.println("AC Volts: ");
    Console.println(Volts);
    Console.println("");
   // Compare temperature & humidity events and perform a check sum.
   if (isnan(TemperatureC) || isnan(Hum)){
    Console.println("Bad Check Sum Value");
     }
  }
}

void updateData() {
  String txt = "";          // Text to send
  String txt1 = "";
  String txt2 = "";
  String txt3 = "";
  txt = String (TempF);
  txt1 = String (Hum);
  txt2 = String (Light);
  txt3 = String (Volts);

  // convert the readings to a String to send it:
 //---------------------------------------------
 dataString = "{\"protocol\":\"v1\",\"checksum\":\"\",\"device\":\"";
  dataString += DEVICE;
 dataString += "\",\"at\":\"now\",\"data\":{\"TempF\":"+txt;
  dataString += ",\"Hum\":"+txt1;
  dataString += ",\"Light\":"+txt2;
  dataString += ",\"Volts\":"+txt3;
  dataString += "}}";
}

void sendData() {
  // form the string for the APIKEY header parameter:
  String apiString = "carriots.apikey: ";
  apiString += APIKEY;

  // Send the HTTP POST request
  // --------------------------
  Process carriots;
  Console.println("Sending data... ");
 carriots.begin("curl");
 carriots.addParameter("-k");
 carriots.addParameter("--request");
 carriots.addParameter("POST");
 carriots.addParameter("--data");
 carriots.addParameter(dataString);
 carriots.addParameter("--header");
 carriots.addParameter(apiString);
carriots.addParameter("https://api.carriots.com/streams/");
  carriots.run();
  if (debug_mode == true){ // Print
```

```
   Console.println("done!");
  }
}
// Add this bottom code to monitor for incoming calls from carriots state changes
//-------------------------------------------------------------------------------
  // See If there's incoming data from the net connection,
  // send it out the Serial:
  //while (carriots.available() > 0) {
    //char c = carriots.read();
    //Serial.write(c);  // Change to console.write?
  //}
```