```
/*
 * Script _all.ino
 * This script runs the sensor package
 * Uses objects for each of the sensors
 * Prints information to Serial screen
 * Publishes data to ThingSpeak
 */

#include "CALCULATE_MRT.h"
#include "MHZ19.h"
#include "CCS821.h"
#include "SHT35D.h"
#include "MRT.h"
#include "PM.h"
#include "Time.h"
#include <Wire.h>

// create instances of objects
PM_7003 myPM;
ClosedCube_Si7051 myMRT;
ClosedCube_SHT31D mySHT;
Adafruit_CCS811 myVOC;
MHZ19 myCO2;
mrt_and_ot my_MRT_OT;

/*
 * Boolean expressions
 * start_xxx indicate whether sensor has been read from properly
 * read_from_xxx indicate whether or not to read from sensor_xxx (changes throughout
 code)
 * finished_xxx indicates whether done reading from a sensor (read a good average)
 */
bool start_co2 = false;
bool start_voc = false;
bool start_sht = false;
bool start_pm = false;
bool start_mrt = false;

bool read_from_co2 = true;
bool read_from_pm = false;

bool finished_co2 = false;
bool finished_pm = false;
bool finished_other_sensors = false;
bool finished_mrt_ot = false;
bool finished_voc = false;

// average reading values
int co2_ave = -1;
float sht_rh_ave = -1;
float sht_t_ave = -1;
float voc_eCO2_ave = -1;
float voc_TVOC_ave = -1;
int pm_ave = -1;
float T_g = -1;
float T_a = -1;
float T_mrt = -1;
float T_ot = -1;

bool publish_data = true; // should we publish data?

// pin numbers for pm and co2 sensors
int pm_transistor_control = A4;
int pm_tx_transistor_control = A5;
int co2_transistor_control = A3;

void setup() {
    /*
     * Start Serial and Wire connections
```

```
     * Initialize transistor control for CO2 and PM
     * Turn CO2 sensor on (make_sensor_read())
     * Test all I2C sensors (MRT, SHT, VOC)
     * Stop and wait for 30 seconds (warm-up)
     */
    Serial.begin(9600);
    Wire.begin();
    Serial.println("Initializing");

    myCO2.set_transistor(co2_transistor_control);
    myPM.set_transistor(pm_transistor_control, pm_tx_transistor_control);

    myCO2.make_sensor_read();

    start_mrt = myMRT.start_mrt();
    Serial.println("-----------------------");

    start_sht = mySHT.start_sht();
    Serial.println("-----------------------");

    start_voc = myVOC.start_voc();
    Serial.println("-----------------------");
    Serial.println("30 second delay");
    Serial.println("-----------------------");
    delay(30000);

}

void loop() {
    /*
     * Wait for CO2 sensor to warm-up (PM sensor is off)
     * Read from MRT, SHT, and VOC sensors while CO2 sensor warms-up
     * After CO2 sensor warms-up, read from CO2 sensor and save average reading
     * Save average value from MRT, SHT, and VOC sensors
     * Turn off CO2 sensor, turn on PM sensor
     * Read from MRT, SHT, and VOC sensors while PM sensor warms-up
     * After PM sensor warms-up, read from PM sensor and push all data to ThingSpeak
     * Repeat
     */

    // Decide which of CO2 or PM sensor to read from
    if(read_from_co2) {
        start_co2 = myCO2.make_sensor_read();
        start_pm = false;

        if(start_co2) {
            read_from_co2 = false;
            read_from_pm = true;
            finished_co2 = true;
        }
    }
    else if(read_from_pm) {
        start_pm = myPM.make_sensor_read();
        start_co2 = false;

        if(start_pm) {
            read_from_pm = false;
            read_from_co2 = true;
            finished_pm = true;
        }
    }

    start_mrt = myMRT.run_mrt(); //read from MRT sensor

    // Read from SHT sensor, or restart SHT sensor
    if(start_sht) {
        Serial.println("Reading from SHT Sensor");
        Serial.println("-----------------------");
        start_sht = mySHT.run_sht();
```

```
        Serial.println("----------------------");
    }
    else if(!start_sht) {
        Serial.println("--------------------------");
        Serial.println("Not reading from SHT Sensor");
        Serial.println("--------------------------");
        Serial.println("Tring to start SHT");
        start_sht = mySHT.start_sht();
        Serial.println("----------------------");
    }
    // Read from VOC sensor, or restart VOC sensor
    if(start_voc) {
        Serial.println("Reading from VOC Sensor");
        Serial.println("----------------------");
        start_voc = myVOC.run_voc();
        Serial.println("----------------------");
    }
    else if(!start_voc) {
        start_voc = myVOC.start_voc();
        Serial.println("Reading from VOC Sensor");
        Serial.println("----------------------");
        start_voc = myVOC.run_voc();
        Serial.println("----------------------");
    }

    // If done reading from CO2 sensor, save CO2, MRT, SHT, and VOC readings
    if(finished_co2 && !finished_other_sensors) {
        finished_other_sensors = true;

        if(!finished_mrt_ot) {
            if(start_mrt && start_sht){
                T_g = myMRT.get_MRT_ave();
                T_a = mySHT.get_t_ave();
                sht_rh_ave = mySHT.get_rh_ave();
                my_MRT_OT.calculate_mrt_and_ot(T_g, T_a);
                T_mrt = my_MRT_OT.get_mrt();
                T_ot = my_MRT_OT.get_ot();
                finished_mrt_ot = true;
            }
            else if(start_mrt && !start_sht) {
                T_g = myMRT.get_MRT_ave();
                T_a = -1;
                sht_rh_ave = -1;
                T_mrt = -1;
                T_ot = -1;
            }
            else if(!start_mrt && start_sht) {
                T_g = -1;
                T_a = mySHT.get_t_ave();
                sht_rh_ave = mySHT.get_rh_ave();
                T_mrt = -1;
                T_ot = -1;
            }
            else {
                T_g = -1;
                T_a = -1;
                sht_rh_ave = -1;
                T_mrt = -1;
                T_ot = -1;
            }
        }

        if(start_voc && !finished_voc){
            voc_eCO2_ave = myVOC.get_eCO2_ave();
            voc_TVOC_ave = myVOC.get_TVOC_ave();
            finished_voc = true;
        } else {
            voc_eCO2_ave = -1;
            voc_TVOC_ave = -1;
```

```
        }

        co2_ave = myCO2.get_co2_ave();

        if(finished_mrt_ot && finished_voc) {
            finished_other_sensors = true;
        }
    }

    // If done reading from PM and CO2 sensors, save PM reading and push to ThingSpeak
    if(finished_co2 && finished_pm) {
        pm_ave = myPM.get_pm_ave();
        finished_co2 = false;
        finished_pm = false;
        finished_mrt_ot = false;
        finished_voc = false;
        finished_other_sensors = false;

        if(publish_data) {
            char data[1000];
            sprintf(data,"{ \"Mean Radiant Temperature\": \"%3.2f\", \"Operating
            Temperature\": \"%3.2f\", \"CO2 Concentration\": \"%i\", \"eCO2\":
            \"%4.2f\", \"TVOC\": \"%4.2f\",\"PM 2_5\": \"%i\", \"Air Temperature\":
            \"%3.2f\",\"Relative Humidity of Air\": \"%3.2f\"}" , T_mrt, T_ot, co2_ave,
            voc_eCO2_ave, voc_TVOC_ave, pm_ave, T_a, sht_rh_ave);
            Serial.println("------------------------");
            Serial.print("Data:");
            Serial.println(data);
            Serial.println("------------------------");

            Particle.publish("IEQ Final Prototype", data, PRIVATE);

            myCO2.reset_co2_ave();
            myPM.reset_pm_ave();
        }
    }
}
```