

# Arduino Temperatuursensor Code

## TemperatuurUitlezenJson\_04.ino

```
#include <Arduino.h>
#include "DHT11_Aggregator.h"

#define DHTTYPE DHT11
const uint8_t pins[] = {7, 8, 9};

DHT11_Aggregator sensorGroup(pins, 3, DHTTYPE);

bool handshake = false;

void setup() {
  Serial.begin(9600);
  sensorGroup.begin();
}

void loop() {
  if (Serial.available() > 0) {
    byte incomingByte = Serial.read();

    if (incomingByte == 0b00100100) {
      Serial.println("ACK");
      handshake = true;
    }

    else if (incomingByte == 0b10100101 && handshake) {
      sensorGroup.update();

      String data = "{\"temperatuur\":";
      data += String(sensorGroup.getTemperature(), 2);
      data += "\",\"luchtvochtigheid\":";
      data += String(sensorGroup.getHumidity(), 2);
      data += "}";
      Serial.println(data);

    } else {
      Serial.println("NACK");
    }
  }

  delay(10);
}
```

# DHT11\_Aggregator.h

```
#ifndef DHT11_AGGREGATOR_H
#define DHT11_AGGREGATOR_H

#include <Arduino.h>
#include <DHT.h>

class DHT11_Aggregator {
public:
    DHT11_Aggregator(const uint8_t pins[], uint8_t count, uint8_t type);
    ~DHT11_Aggregator();

    void begin();
    void update();
    float getTemperature() const { return avgTemperature; }
    float getHumidity() const { return avgHumidity; }

private:
    DHT** sensors;          // array van pointers
    uint8_t sensorCount;
    float avgTemperature;
    float avgHumidity;
    uint8_t sensorType;
};

#endif
```

# DHT11\_Aggregator.cpp

```
#include "DHT11_Aggregator.h"

DHT11_Aggregator::DHT11_Aggregator(const uint8_t pins[], uint8_t count,
uint8_t type)
    : sensorCount(count), avgTemperature(0), avgHumidity(0),
    sensorType(type) {
    sensors = new DHT*[sensorCount]; // array van pointers, deze begrijp
ik nog slecht. waarom DHT* en niet DHT? Het werkt zo wel..
    for (uint8_t i = 0; i < sensorCount; i++) {
        sensors[i] = new DHT(pins[i], sensorType); // maak elk object
individueel aan
    }
}

DHT11_Aggregator::~DHT11_Aggregator() {
    for (uint8_t i = 0; i < sensorCount; i++) {
        delete sensors[i];
    }
}
```

```
    }  
    delete[] sensors;  
}  
  
void DHT11_Aggregator::begin() {  
    for (uint8_t i = 0; i < sensorCount; i++) {  
        sensors[i]->begin();  
    }  
}  
  
void DHT11_Aggregator::update() {  
    float tempSum = 0;  
    float humSum = 0;  
  
    for (uint8_t i = 0; i < sensorCount; i++) {  
        tempSum += sensors[i]->readTemperature();  
        humSum += sensors[i]->readHumidity();  
    }  
  
    avgTemperature = tempSum / sensorCount;  
    avgHumidity = humSum / sensorCount;  
}
```