

senseBox:home

Dokumentation



Inhaltsverzeichnis

Einleitung	1.1
Inventar & Grundaufbau	1.2
Softwareinstallation	1.3
openSenseMap Registrierung	1.4
Beispielaufbauten	1.5
senseBox fibox	1.5.1
senseBox:home mit ESP8266	1.6
Softwareinstallation	1.6.1
Anpassen des Sketches	1.6.2
senseBox mit Wemos D1	1.6.3
Alle weiteren ESP8266	1.6.4



senseBox:home

Die senseBox:home ist ein Citizen Science DIY-Toolkit für die ortsbezogene Messung von Umweltdaten wie Temperatur, Luftfeuchte, Luftdruck, Beleuchtungsstärke und UV-Licht. Sie basiert auf der Arduino/Genuino Plattform und kann einfach in unsere Sensorweb-Plattform [openSenseMap](#) integriert werden, wo sie kontinuierlich Messdaten liefert.

Auf diesen Seiten - welche auch als [PDF](#) verfügbar sind - befindet sich die Dokumentation und Aufbauanleitung zur senseBox:home.

Wie die Sensorstation programmiert wird, ist in den folgenden Kapiteln beschrieben:

Bei Fragen zum Aufbau wende dich bitte [per Mail](#) an uns. Das senseBox Team wünscht viel Spaß mit deiner Do-It-Yourself Sensorstation!

Die senseBox ist ein Open Source Projekt und befindet sich ständig in der Weiterentwicklung. Das heißt, dass auch diese Seiten nach und nach erweitert werden. Falls euch Fehler auffallen oder ihr bei der Entwicklung einsteigen wollt, meldet euch gerne bei uns [per Mail](#) oder [öffnet ein Issue auf GitHub!](#)

Warnhinweise:

- Durch elektrostatische Entladung können die Bauteile beschädigt oder sogar zerstört werden! Daher solltet ihr euch z.B. an einem Heizungsrohr entladen, bevor ihr mit dem Aufbau anfängt.
- Elektronische Bauteile und Leiterplatten können Chemikalien enthalten. Daher solltet ihr nach dem Aufbau oder Gebrauch euch die Hände waschen.
- Elektronik sollte umweltfreundlich entsorgt werden und bei Sammelstellen abgegeben werden.

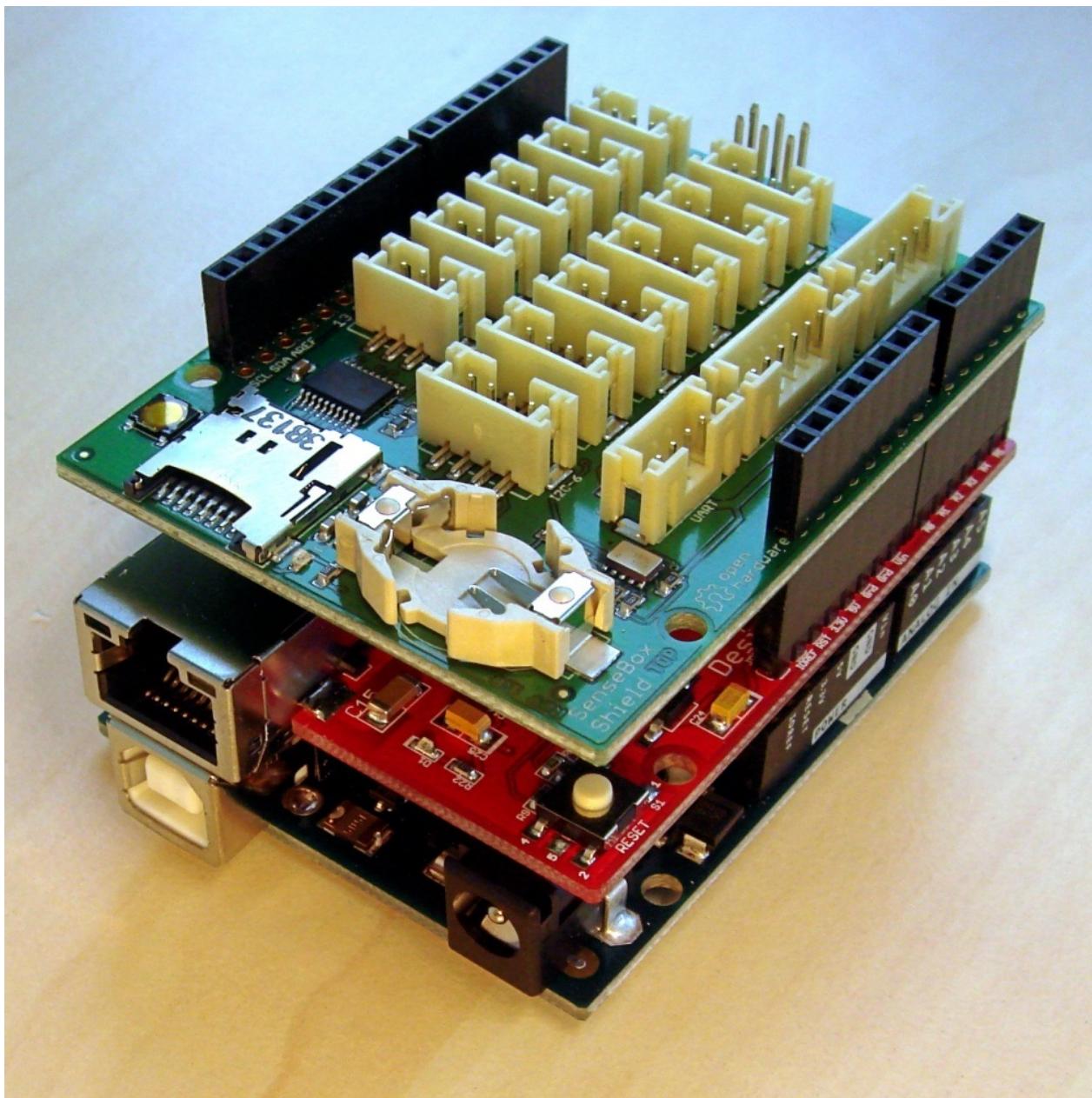
Inventarliste

Bevor es los geht solltet ihr überprüfen ob alle Bauteile vorhanden sind.

Inhalt der senseBox

Basisstation bestehend aus drei Platinen

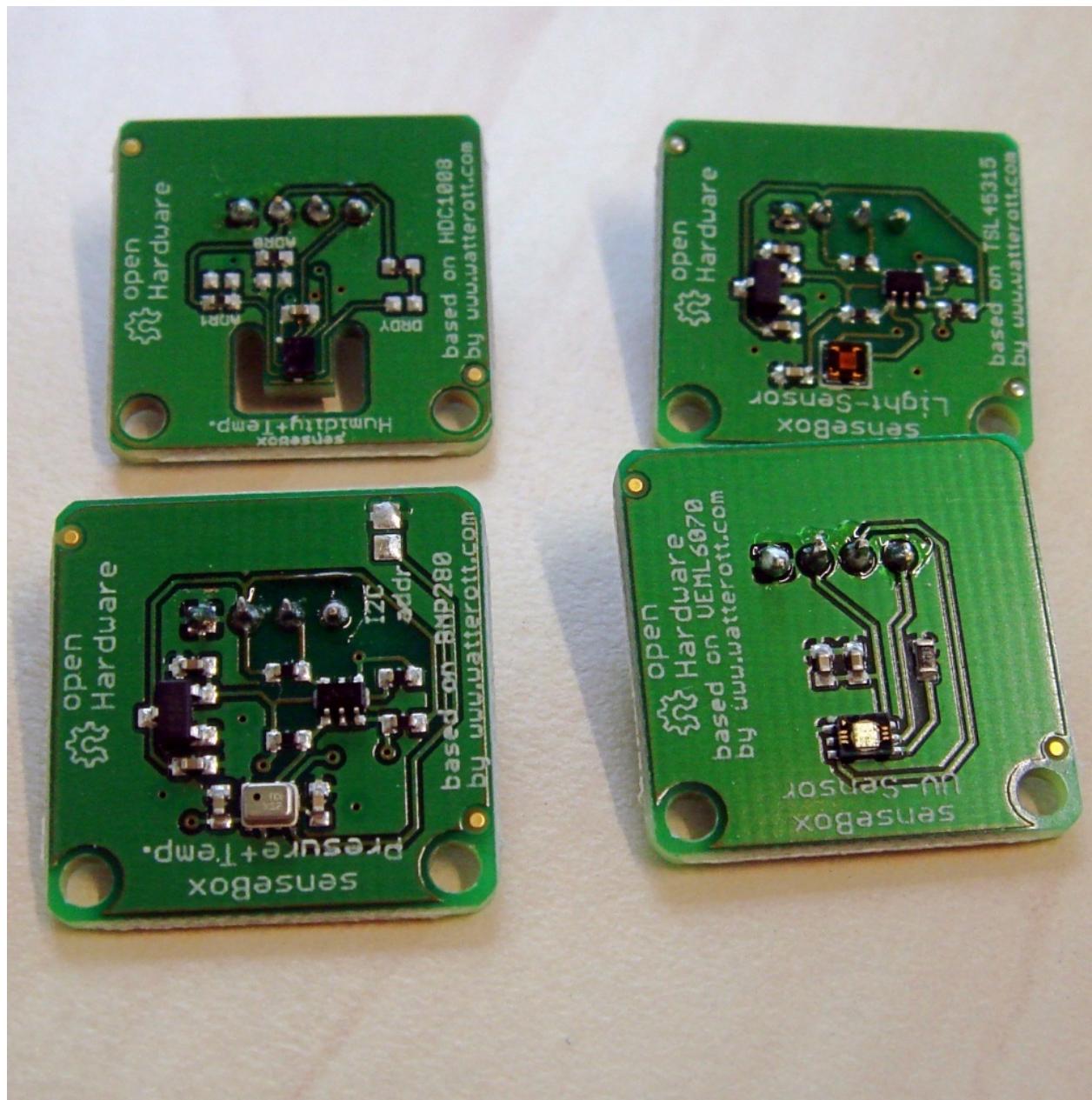
Die senseBox:home ist in zwei Ausgaben verfügbar: Einmal mit LAN-, und einmal mit WLAN-Netzwerkverbindung. Je nach Ausgabe ist ein W5500 Ethernet Shield, oder ein Watterott WLAN-Shield enthalten.



Platine	Beschreibung
Genuino Uno (unten)	Liest die angeschlossenen Sensoren aus und überträgt die Messungen ins Internet
W5500 Ethernet Shield oder Watterott WLAN-	Ist für die Internetverbindung zuständig

Shield (mitte)	Ist für die Internetverbindung zuständig
senseBox Shield (oben)	Hier werden die Sensoren angeschlossen

Grundausstattung mit vier Sensoren



Sensor	Beschreibung
HDC1008	Temperatur in Grad Celsius (°C) und relative Luftfeuchte in Prozent (%)
BMP280	Luftdruck in Pascal (pa)
TSL45315	Beleuchtungsstärke des sichtbaren Lichts in Lux (lx)
VEML6070	Intensität der ultravioletten Strahlung in Mikrowatt pro Quadratcentimeter ($\mu\text{W}/\text{cm}^2$)

Anschlusskabel für Sensoren und USB-Verbindung

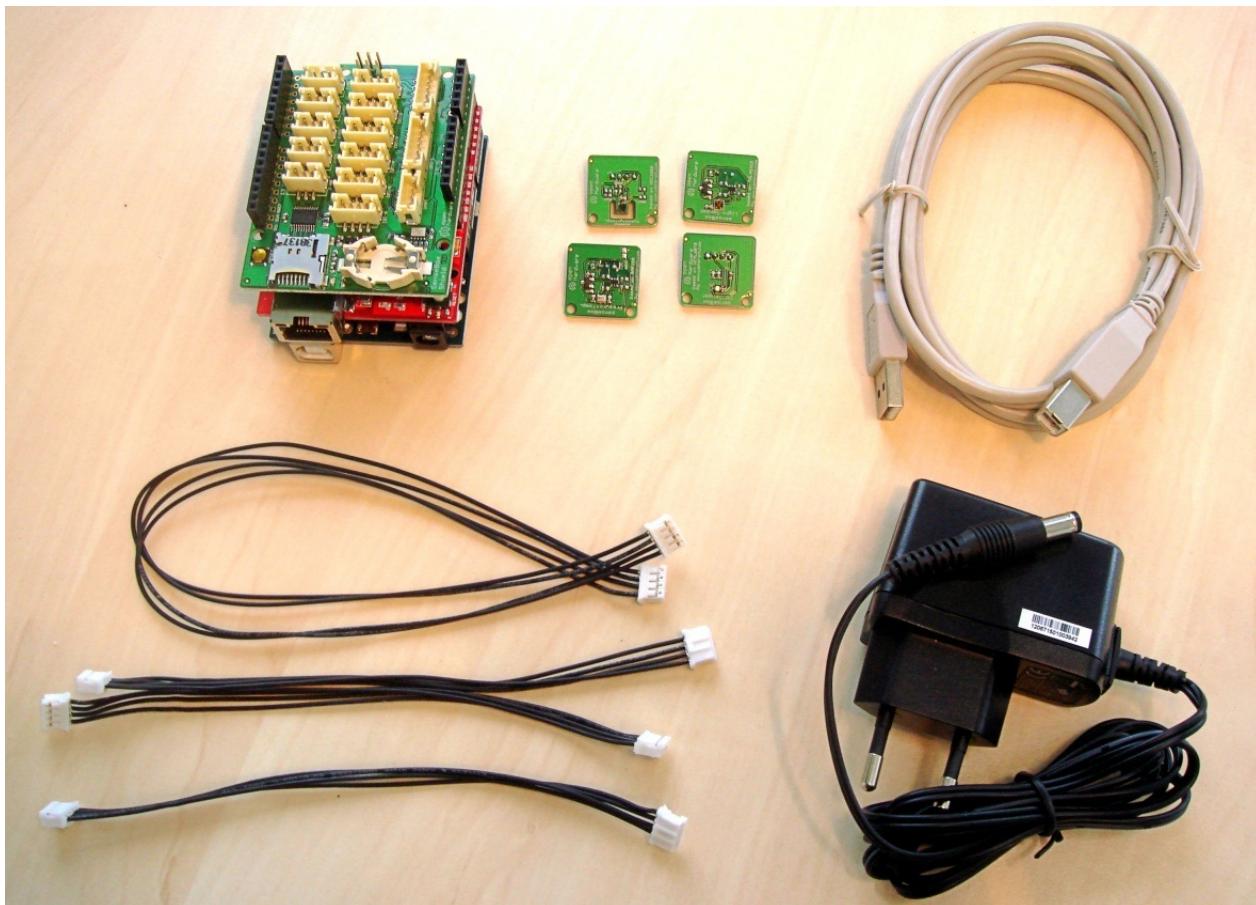
- 1x USB-Kabel für den Anschluss des Mikrocontrollers an den Computer
- 1x langes Verbindungskabel für kombiniertes Thermo- bzw. Hygrometer

- 3x kurzes Verbindungskabel für Barometer, Luxmeter und UV-Lichtsensor

Netzteil

- 9V Netzteil (670mA)

Gesamtüberblick:



Zusätzliche Materialien (NICHT im Lieferumfang enthalten)

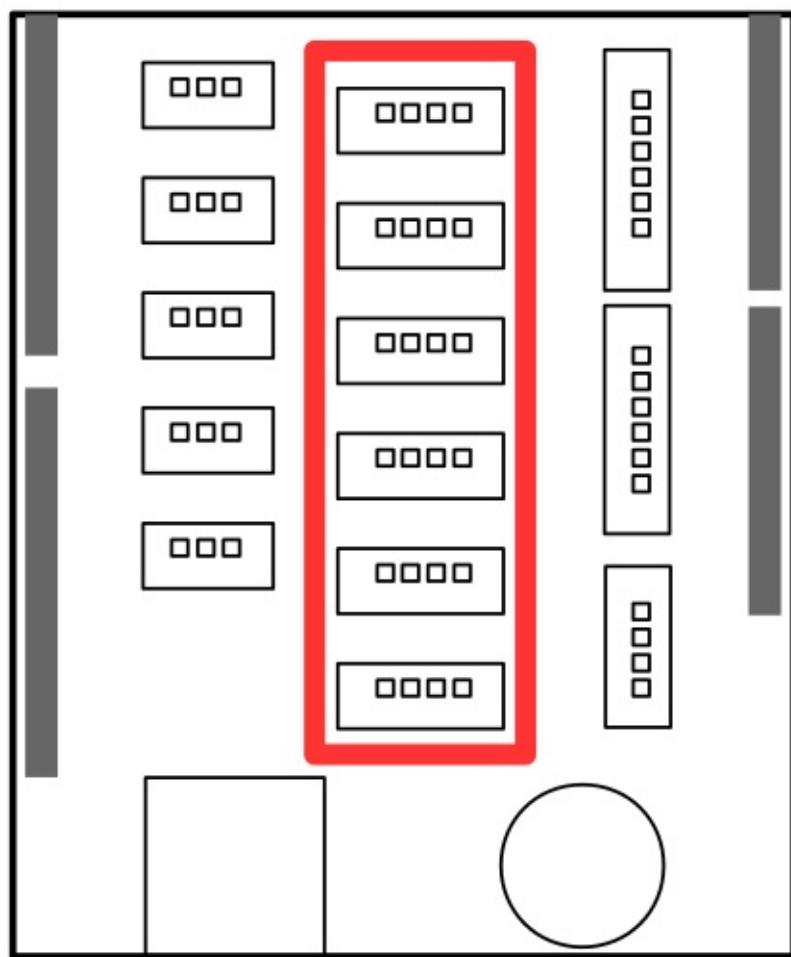
- LAN-Kabel für den Anschluss der senseBox an euren Router, falls die senseBox:home LAN vorliegt
 - Gehäuse für eine wetterfeste Installation der Elektronik
 - Werkzeuge für den Aufbau wie z.B. Heißklebepistole
-

Aufbau der senseBox

Du kannst deine Messstation kann in wenigen Schritten zusammenbauen.

Die senseBox wird entweder über das USB-Kabel oder über das Netzteil mit Strom versorgt. Für den temporären Betrieb wird das Netzteil also nicht benötigt.

Im Bausatz der senseBox:home befinden sich vier kleinen Platinen mit den Sensoren. Die eigentlichen Sensoren sind nur wenige Millimeter groß und befinden sich auf der Oberseite der Platinen. Um einer Beschädigung vorzubeugen, solltest du die kleinen Sensoren nicht berühren, sondern die Platinen nur am Rand anfassen. Der Anschluss der Sensoren ist denkbar einfach: Benutze die Verbindungskabel, um die Sensoren mit den mittleren Steckplätzen auf der Basisstation zu verbinden. Welcher Anschluss dabei gewählt wird spielt keine Rolle.



Das lange Verbindungskabel ist für den HDC1008 gedacht, um ihn außerhalb eines Gehäuses anbringen zu können!

Treiber und Softwareinstallation

Bevor die senseBox aktiviert werden kann, musst du Treiber sowie eine Software auf deinem Computer installieren. Außerdem ist es vor Inbetriebnahme der senseBox ratsam einen Testlauf durchzuführen, um zu überprüfen ob die Sensoren korrekt funktionieren und die Kommunikation mit dem Internet reibungslos läuft.

Falls etwas bei dem Testlauf schief geht, melde dich am besten [per Mail](#) bei unserem Support.

Arduino Software herunterladen

Für einen reibungslosen Ablauf bitte Arduino 1.6.5 oder höher nutzen.

Das Mainboard der senseBox ist eine modifizierte Version des Arduino Uno Mikrocontrollers. Um ein Programm auf das Board zu laden, brauchst du die integrierte Entwicklungsumgebung von Arduino, kurz Arduino IDE. Lade die neueste Version als zip-Datei von der [Arduino Homepage](#) herunter:



Arduino ist ein Open-Source Projekt und wird durch Spenden finanziert. Daher wirst du vor dem Download nach einer Spende gefragt; das kannst du überspringen indem du auf „just download“ klickst.

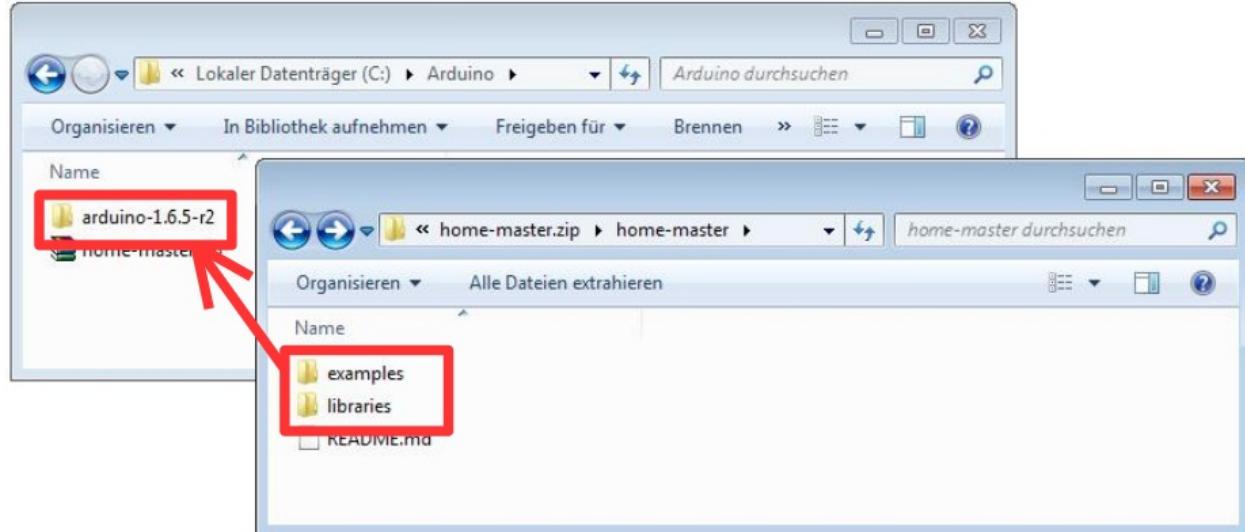


Legt auf eurer Festplatte einen neuen Ordner an und entpackt darin die Zip-Datei.

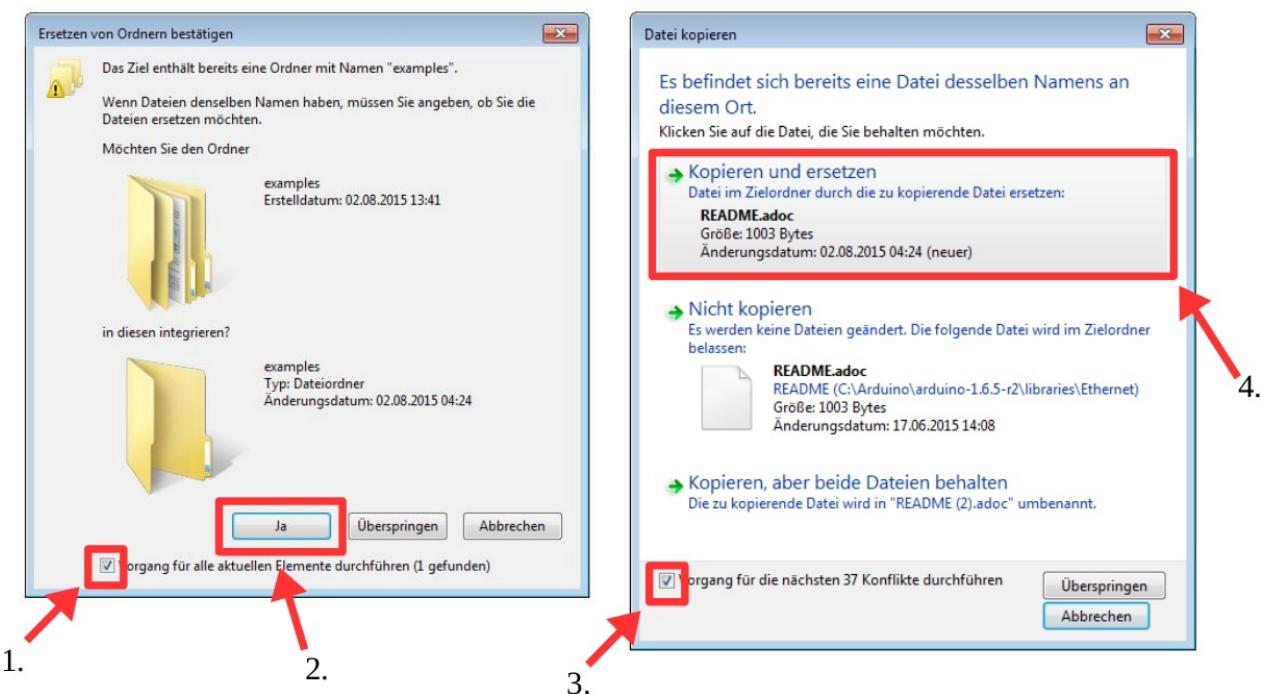
Arduino Bibliotheken installieren

Um die Sensoren und die Netzwerkkarte nutzen zu können, müssen noch ein paar Bibliotheken installiert werden. Ein zip-Archiv mit allen benötigten Bibliotheken findest du [hier](#).

Lade dieses zip-Archiv herunter und integriere nun die beiden Ordner `examples` und `libraries` aus dem Archiv in deinen Arduino Ordner. Wenn ihr gefragt werdet ob bestehende Dateien ersetzt werden sollen, folge den Anweisungen unten auf der Seite.



Setze nun wie unten dargestellt im ersten Dialogfeld den Haken unten und bestätigt mit „Ja“. Daraufhin öffnet sich ein neues Fenster, in dem du wieder den Haken setzt, und „Kopieren und ersetzen“ auswählst.



Das folgende Video zeigt den Kopiervorgang noch einmal im Detail:

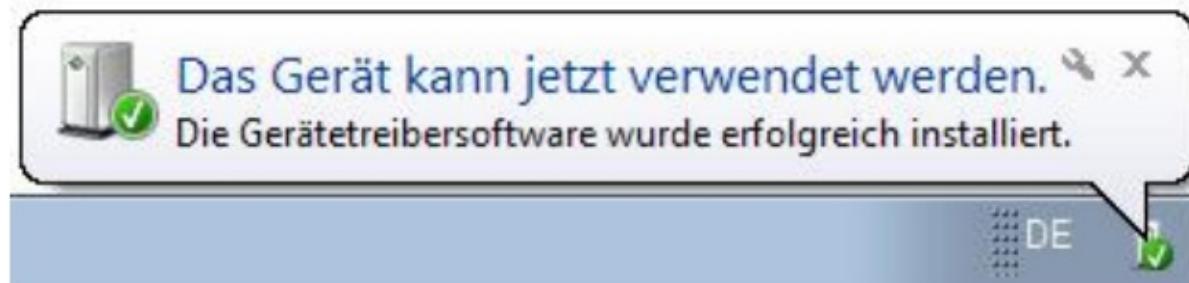
Eingebettetes Video: <https://www.youtube.com/watch?v=j-hdRJp2o4k>

Treiber installieren

Als letzten Schritt für die Softwareinstallation müssen Windows-User einen Treiber installieren. Die senseBox:home gibt es mit verschiedenen Microkontroller Boards. Unten rechts im Bild ist der Kontroller der neuen Version abgebildet (Genuino Uno), links im Bild das Board der alten Version (Wattuino Uno). Bei den neuen Versionen unserer Bausätze welche das Genuino Uno Board enthalten, ist eine Treiberinstallation nicht mehr notwendig. Besitzer der Bausätze mit Wattuino Uno Board müssen diesen Schritt jedoch ausführen!



Die Treiberinstallation sollte bei vorhandener Internetverbindung automatisch funktionieren (getestet mit Win7/Win8/Win10). Dazu einfach den Microkontroller mit dem USB Kabel mit dem Rechner verbinden und abwarten bis die Treiber installiert sind. Dieser Vorgang kann dann bis zu 10 Minuten dauern. Unten rechts in der Taskleiste wird dann entsprechende Meldung erscheinen, sobald die Installation beendet wurde:



Sensoren testen

Die Sensoren der senseBox:home können nun auf Funktionstüchtigkeit und getestet werden. Hierzu gibt es einen Sketch, welcher mit den zuvor Arduino-Bibliotheken installiert wurde. Voraussetzung für den Test ist natürlich, dass die Sensoren mit dem Arduino verbunden wurden ;)

1. Arduino Anwendung starten
 - Es kann sein, dass nach dem Start eine Meldung über neue Updates erscheint. Fall du die Version 1.6.5 oder höher installiert hast, kannst du dies überspringen.
2. unter Werkzeuge → Board das Arduino Uno auswählen
3. unter Werkzeuge → COM-Port den entsprechenden Anschluss wählen
 - Falls mehrere Auswahlmöglichkeiten angezeigt werden, musst du zuerst den richtigen COM-Port im Geräte Manager finden, oder alle Ports ausprobieren.

Ladet nun das Programm, um die Sensoren zu testen und übertragt es auf die Messstation:

1. in der Menüleiste Datei → Beispiele → senseBox → _01_sensor_test auswählen
2. das Programm über das Pfeil Icon auf den Mikrocontroller laden
3. warten bis das Programm übertragen wurde
4. den seriellen Monitor über das Lupe Icon öffnen

```

Starte Sensortest...

Luftfeuchte:      50.90 %
Temperatur:       22.05 C
Luftdruck:        1013.80 hPa
Beleuchtungsstaerke: 390.00 lx
UV-Strahlung:     45 uW/cm2

Luftfeuchte:      51.00 %
Temperatur:       22.05 C
Luftdruck:        1013.74 hPa
Beleuchtungsstaerke: 10.00 lx
UV-Strahlung:     0 uW/cm2

```

Autoscroll Kein Zeilenende 9600 Baud

Du kannst durch experimente prüfen, ob sich die gemessene Temperatur, Luftfeuchtigkeit oder Beleuchtungsstärke verändern. Der Luftdruck lässt sich nicht ohne weiteres beeinflussen. Er sollte grob, je nach Höhenlage und Wetterverhältnissen, zwischen 600 hPa und 1000 hPa liegen. Die Intensität des UV - Lichts kann nur mit speziellen Lampen oder durch direkte Sonneneinstrahlung getestet werden. In einem geschlossen Raum sollte keine bzw. nur minimale UV-Strahlung gemessen werden können.

Verbindung zur openSenseMap testen

Nach dem selben Schema kann noch die Internetverbindung der senseBox:home getestet werden:

1. den seriellen Monitor (Fenster mit den Messwerten) schließen
2. ein Netzwerkkabel von eurem Heimnetzwerk mit der senseBox verbinden
3. in Menüleiste Datei → Beispiele → senseBox → _02_network_test auswählen
4. das Programm über das Pfeil Icon auf den Mikrocontroller laden
5. den seriellen Monitor über das Lupe Icon starten

Wenn die Verbindung klappt, bekommst du eine entsprechende Meldung im seriellen Monitor angezeigt.

```

Teste Internetverbindung...verbunden!

HTTP/1.1 200 OK
Date: Sun, 02 Aug 2015 13:55:28 GMT
Server: Apache/2.2.25 (Win32)
Last-Modified: Thu, 30 Apr 2015 11:53:01 GMT
ETag: "600000042850-30da-514efbcd8dc7e"
Accept-Ranges: bytes
Content-Length: 12506
Connection: close
Content-Type: text/html

```

Autoscroll Kein Zeilenende 9600 Baud

Damit die Datenübertragung funktioniert darf Port 8000 nicht von deinem Router geblockt werden. Im Normalfall ist dieser Port aber freigegeben.

Web-Integration

Hier wird die Einbindung der senseBox in unser Sensornetzwerk durch die Registrierung auf der openSenseMap beschrieben.

Registrierung auf der openSenseMap

Unter <https://opensensemap.org/register> kann eine neue Sensorstation für das openSenseMap Sensornetzwerk registriert werden. Bei Schritt 3 der Registrierung wirst du nach einem Hardware Setup gefragt. Wähle dort die „senseBox:home“ aus und setze danach je nach Ausgabe den Haken bei „senseBox:home (Ethernet)“ oder „senseBox:home (WLAN)“.

Ein Software-Programm für einen Arduino Mikrocontroller ist an der Dateiendung `.ino` zu erkennen. Eine solche Datei wird benötigt, um eure senseBox mit der openSenseMap im Internet zu verbinden. Den passenden Sketch bekommst du zusammen mit einer E-Mail zugeschickt, wenn die Station auf der openSenseMap registriert wurde.

Programm auf die Station laden

Nachdem du den `.ino` Anhang der Email heruntergeladen hast, muss dieses Programm auf deine senseBox geladen werden. Wie man genau ein Programm auf den Mikrocontroller lädt, ist [hier](#) ausführlich erklärt worden. Hier die Schritte in der Übersicht:

- Arduino Anwendung öffnen
- In der Menüleiste `Datei` → `Öffnen` auswählen und die `sensebox.ino` Datei auswählen
- Dialog mit "Ja" bestätigen
- Das Programm über das Pfeil Icon auf den Mikrocontroller laden
- Warten bis das Programm übertragen wurde

Wenn alles richtig gelaufen ist, kannst du nun auf der openSenseMap deine Station auswählen und verfolgen, wie Messungen kontinuierlich übertragen werden!

Beispielaufbauten und Bauanleitungen

Hier sind verschiedene exemplarische Aufbauten der senseBox:home dokumentiert. Da wir zur Zeit noch an einer Lösung für ein Gehäuse für den Outdoor-Betrieb arbeiten, musst du diesbezüglich selber kreativ werden. Nutze die folgenden Anleitungen als Empfehlung und Inspiration!

Titel	Autor	Datum
Fibox Gehäuse mit PoE	senseBox Team	2015
MyOSD WiFi (PDF)	MyOSD	Okt. 2016

Wenn du selber einen Aufbau entwickelt und dokumentiert hast, fügen wir diesen gern zu dieser Liste hinzu! Erstelle hierzu einen Pullrequest auf [github](#), oder sende uns die Dokumentation [per Mail](#) (idealerweise im [Markdown Format](#)) zu!

Hinweis: Ein Gehäuse speziell für die senseBox:home befindet sich noch in der Entwicklung. Der hier gezeigte Aufbau ist daher nur ein Beispiel dafür, wie sich die Komponenten anbringen lassen. Beim Aufbau ist auch immer etwas Kreativität gefragt um die Station individuell an den Aufstellungsort anzupassen!

Beispielaufbau

Dieser Beispielaufbau setzt voraus, dass die senseBox vorher auf der openSenseMap registriert und programmiert wurde (siehe [hier](#)).

Um die senseBox draußen aufzubauen, braucht ihr noch ein ausreichend langes Netzwerkkabel, sowie ein wasserfestes Gehäuse. Bei dem Gehäuse müsst ihr darauf achten, dass es einen transparenten Deckel ohne Lichtfilterwirkung hat, damit sinnvolle Lichtmessungen gemacht werden können. Zusätzliche Kosten für ein Gehäuse belaufen sich auf ca. 20€.

Zudem braucht ihr noch eine Heißklebepistole, Bohrmaschine, Schraubendreher sowie ein paar Kabelbinder zur Befestigung.

Im folgenden Video wird der Zusammenbau gezeigt, weiter unten folgt eine Schritt-für-Schritt Anleitung:

Eingebettetes Video: <https://www.youtube.com/watch?v=TgEQvMPjrMA>

Gehäuseaufbau

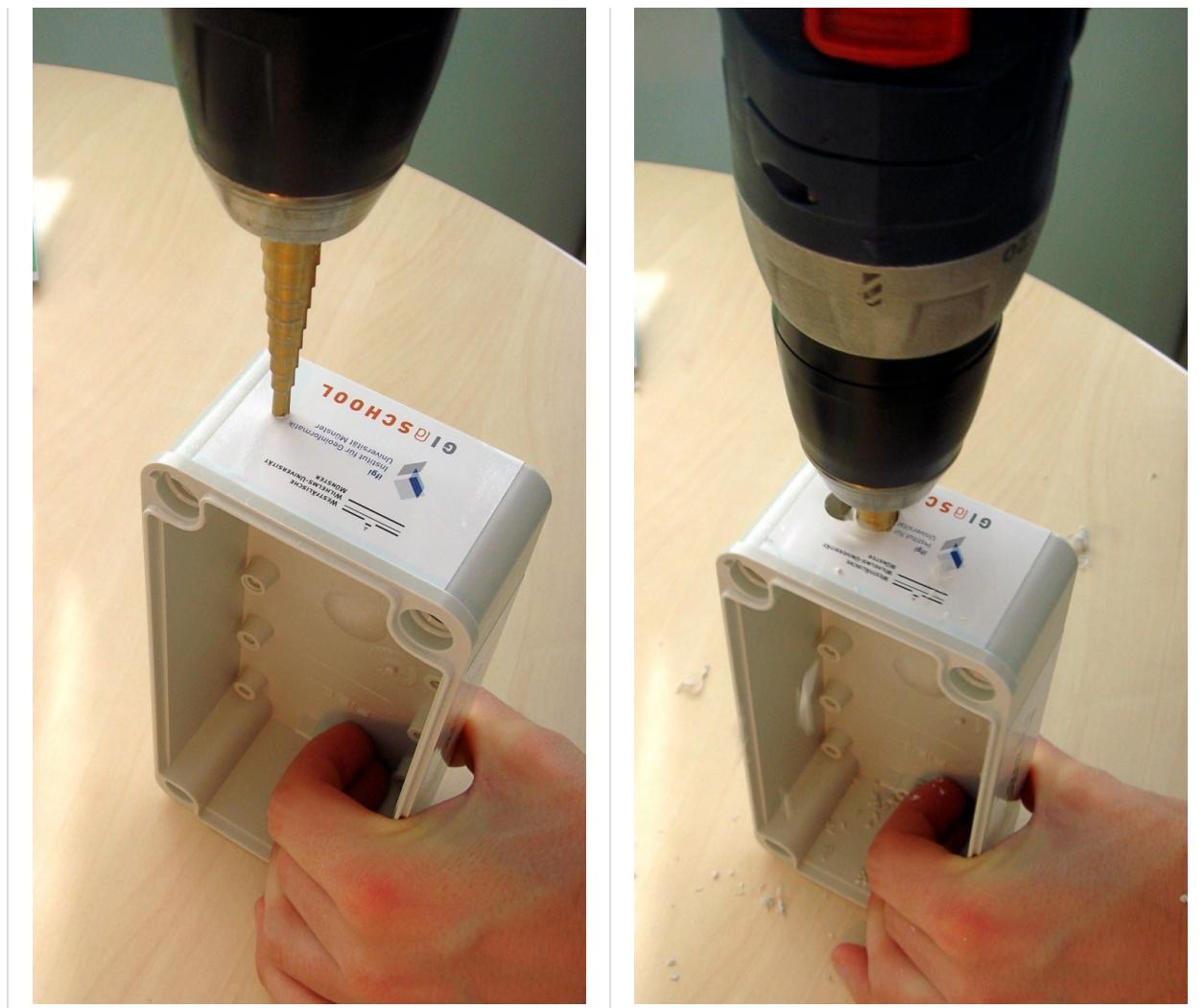
Für unseren Testaufbau benutzen wir [dieses Gehäuse](#) der Firma FIBOX ([hier](#) bestellbar):



Kabelführung bohren

Durch eine ca. 15mm breite Bohrung im Boden des Gehäuses werden Strom- und Netzwerkkabel gelegt, sowie das lange Verbindungs kabel für den HDC1008 Temperatur-/Luftfeuchtesensor:

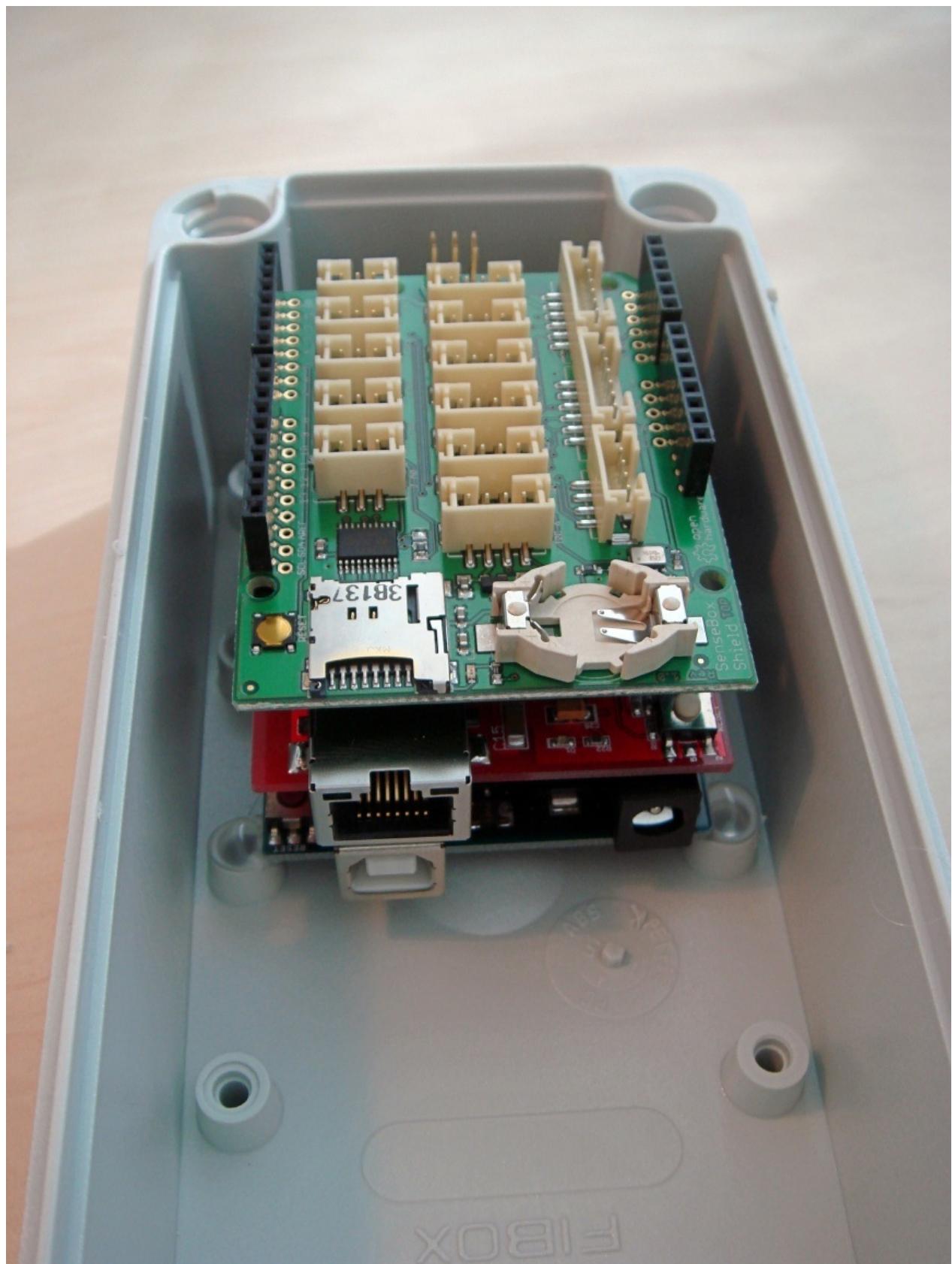




Im Video an der Stelle [00:00](#)

Hauptplatten festkleben

Dazu setzen wir im Gehäuse einige Klebepunkte mit dem Heißkleber und drücken die Hauptplatine an, bis der Kleber getrocknet ist:

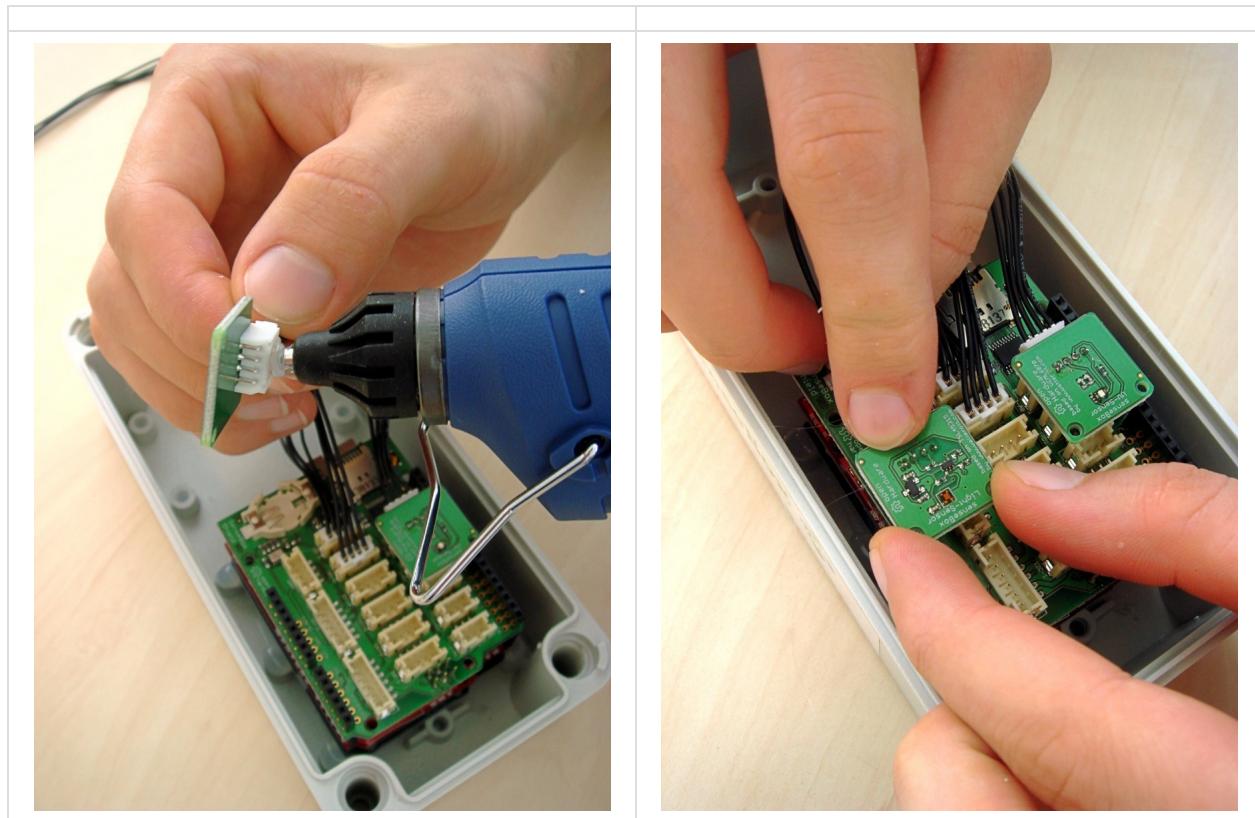


Im Video an der Stelle [00:50](#)

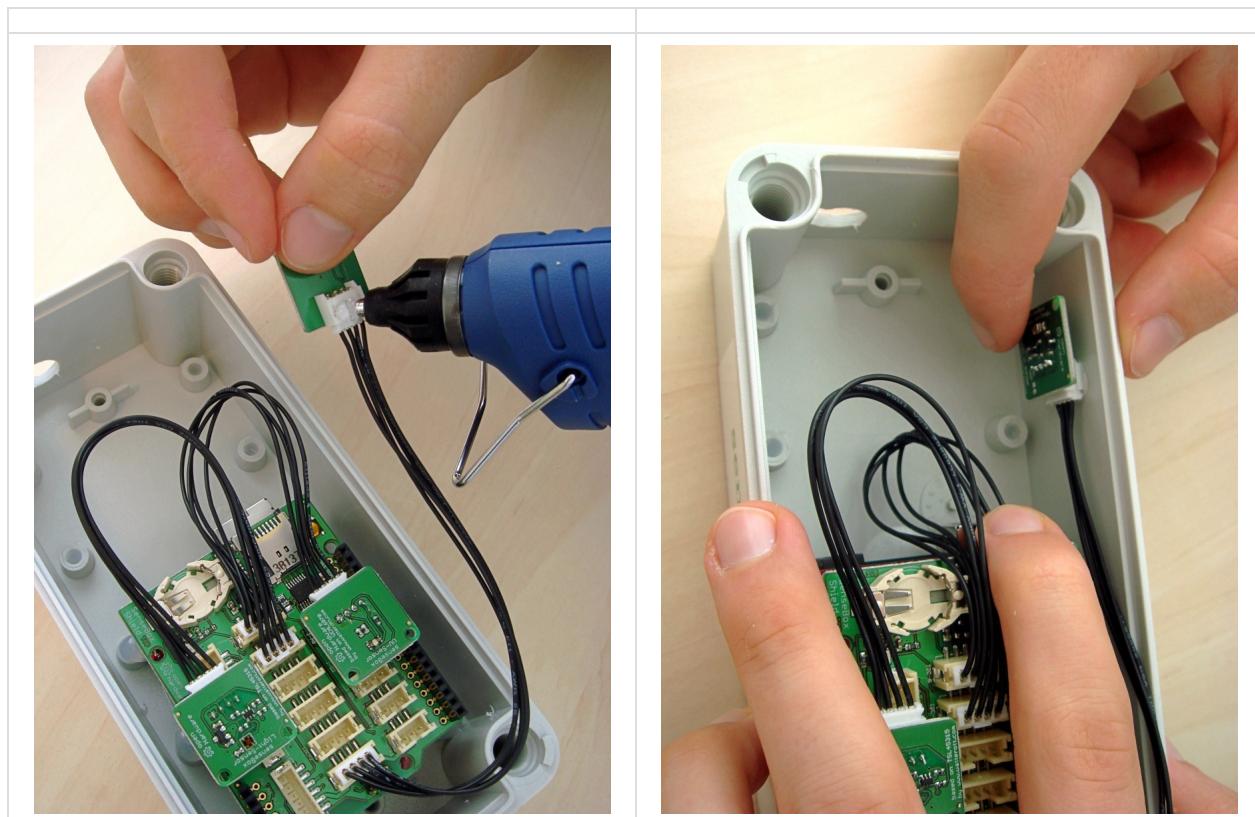
Sensoren befestigen

Es sollte darauf geachtet werden, dass keine Kleberreste auf die Oberseite der Sensorplatten kommt! Beim festkleben der Sensoren reicht schon ein wenig Heißkleber aus.

Die beiden Lichtsensoren oben auf das senseBox - Shield kleben. Die beiden Lichtsensoren sollten "freie Sicht" zum transparenten Deckel haben und nicht von den Kabeln bedeckt werden!



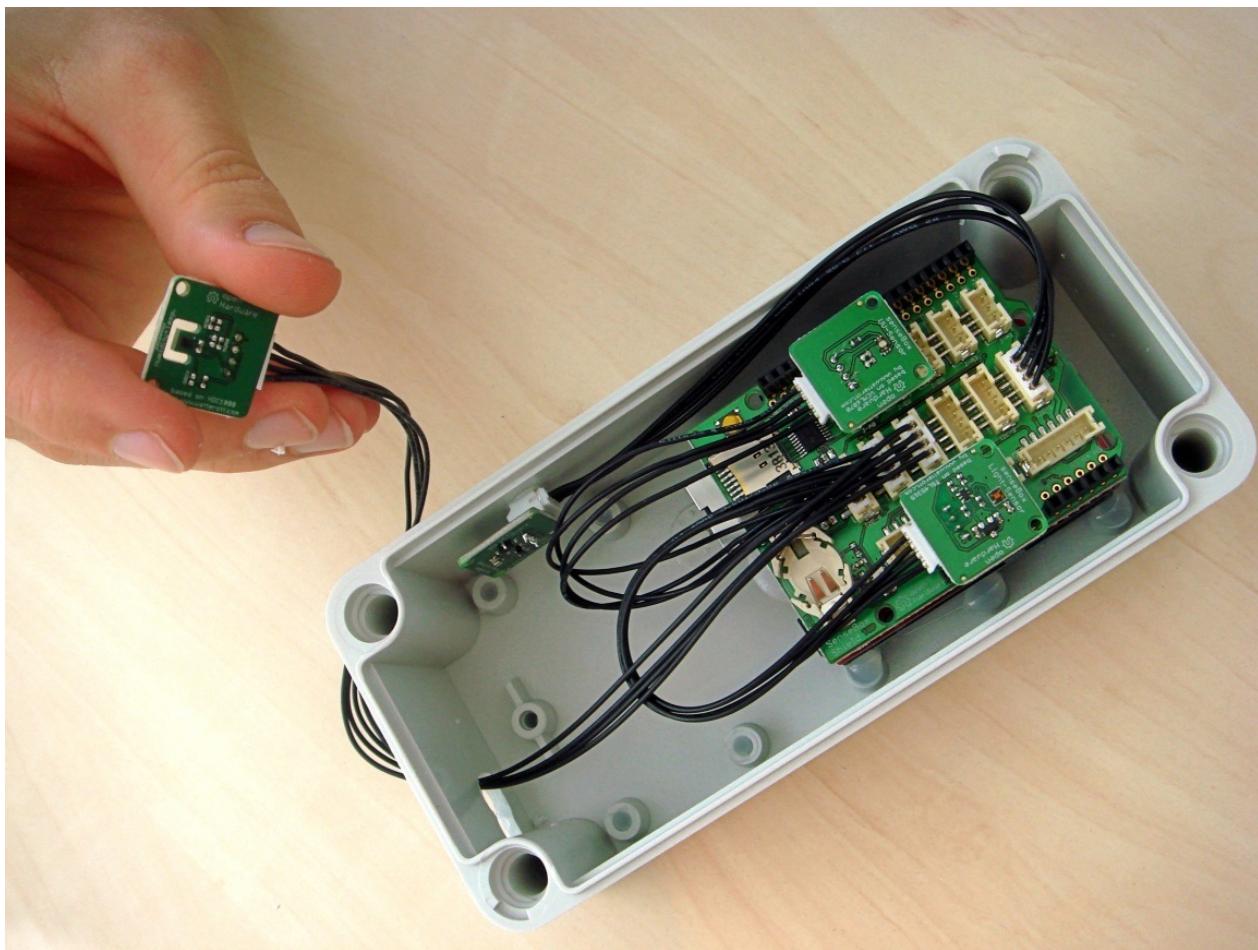
Den Luftdrucksensor ebenfalls im Gehäuse mit etwas Heißkleber weiter vorne befestigen:



Im Video an der Stelle **02:02**

Temperatursensor anbringen

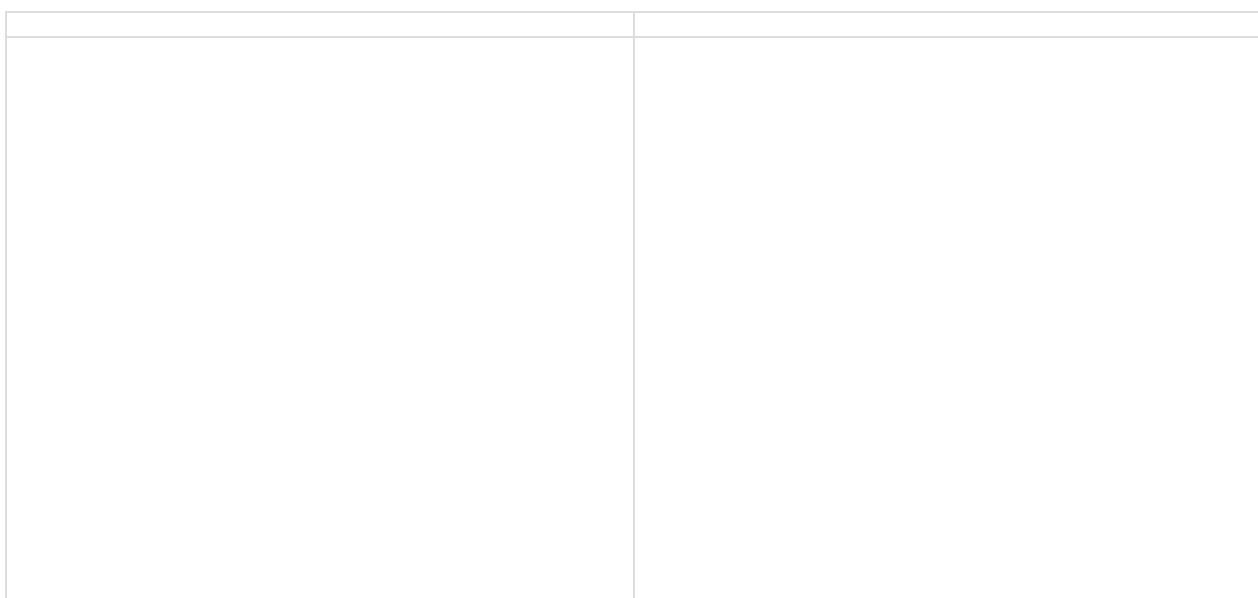
Im Gehäuse werden Temperatur und Luftfeuchte durch die Eigenwärme des Mikrocontrollers beeinflusst. Daher muss der HDC1008 außerhalb in einem zweiten Gehäuse angebracht werden, in dem er vor Regen oder Spritzwasser geschützt ist. Dazu führen wir das lange Sensor-Verbindungskabel durch die Bohrung nach außen und verbinden es mit dem Sensor.

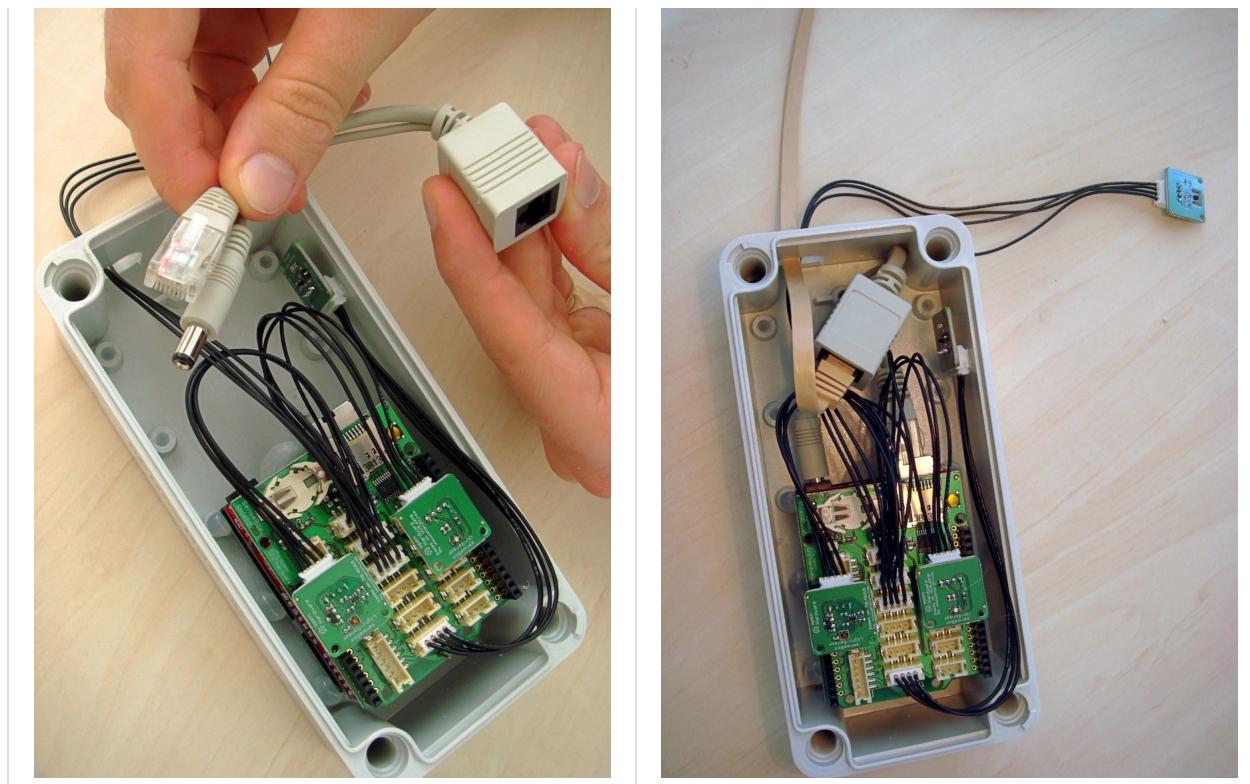


Im Video an der Stelle [04:20](#)

Strom- und Netzwerkanschluss

Um die senseBox mit Strom zu versorgen, kann ein Power-over-Ethernet-Adapter (POE) verwendet werden. Dieser wird an den Netzwerk- und Stromanschluss der Hauptplatinen angeschlossen. Danach kann das Ethernetkabel durch das Bohrloch geführt und in den Adapter gesteckt werden.



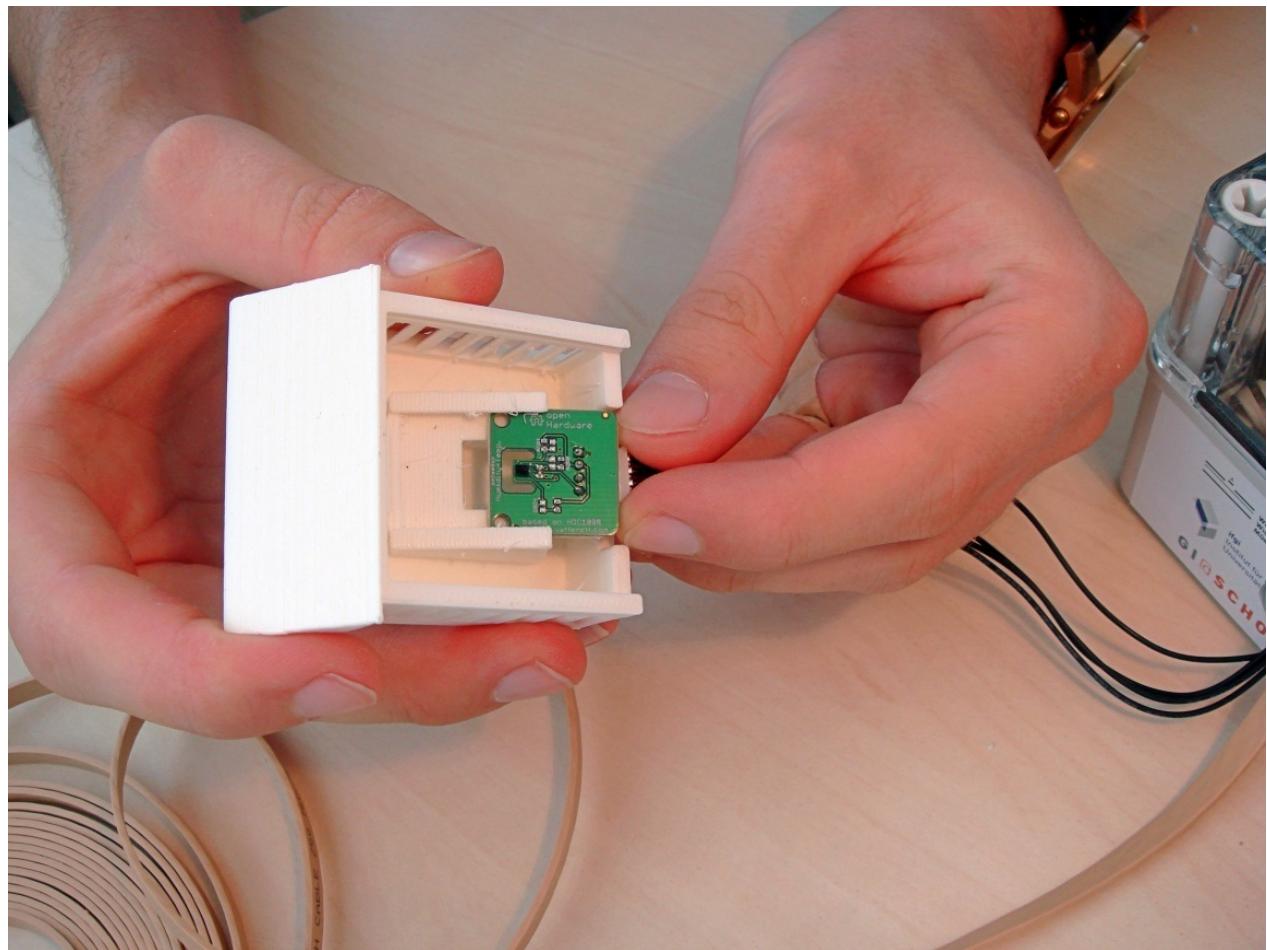


Im Video an der Stelle [05:00](#)

Nun kann das Gehäuse verschlossen werden.

Temperatursensorgehäuse

Damit der Temperatur- und Luftfeuchtesensor durch Regen und Schmutz geschützt ist, wird er in einem eigenen Gehäuse untergebracht. Dazu den Sensor einfach in das 3D-gedruckte Gehäuse schieben und danach die Klappe aufschieben.



Das Gehäuse kann danach etwa mit Heißkleber an das Haupt-Gehäuse geklebt werden.

Im Video an der Stelle [07:12](#)

Kabelzugang abdichten

Damit kein Wasser durch die Kabelführung ins Innere des Gehäuses eindringt, muss diese Öffnung abgedichtet werden. Dazu eignet sich beispielsweise Dichtungsmasse oder auch Heißkleber:



Im Video an der Stelle [08:08](#)

Endergebnis

Zuletzt noch den zweiten POE-Adapter mit dem Ende des Ethernetkabels, und diesen mit Router und Netzteil verbinden.

fertig!



Sucht euch einen Standort für die Station, an dem ihr eure Daten aufnehmen wollt. Im besten Falle sollte dieser Standort exponiert unter freiem Himmel sein. Da die Lage der Station allerdings durch Länge der Kabel begrenzt ist, werden die privaten Stationen in der Regel auf einem Balkon oder an einer Häuserwand befestigt.

Achtung: Falls der Stromstecker nach draußen verlängert werden muss, ist unbedingt darauf zu achten, dass eine witterfeste Verteilerdose verwendet wird! Diese sollte mindestens die **Schutzart IP65** erfüllen.

senseBox:home mit ESP8266

Alternativ zum Arduino/Genuino lassen sich auch andere Plattformen als Microcontroller der Sensorstation nutzen. Hier wird beispielhaft der ESP8266 verwendet, welcher ein schnellerer Mikrokontroller mit eingebautem WLAN ist. Er lässt sich mit der Arduino IDE programmieren, was die Anpassungsarbeit der Software sehr einfach macht. Es gibt relativ viele Varianten dieses Mikrokontrollers.

Achtung

- Leider funktioniert das Barometer (BMP280) nicht korrekt mit der `BMP280.h` Bibliothek, stattdessen kann die `BME280.h` Bibliothek verwendet werden.
- Diese Anleitung beschreibt die Möglichkeit den `ESP8266` mit der Arduino IDE zu programmieren. Es existieren jedoch viele weitere Möglichkeiten Software für den ESP8266 zu schreiben.
- Der ESP8266 kann nur mit 3.3V betrieben werden. Der 3.3V Pin des Arduinos stellt jedoch nicht genügend Strom bereit, um den ESP zu betreiben, eine externe Stromversorgung ist nötig!

1. Alle ESP8266 Varianten: [Installation der Software](#)
2. Alle ESP8266 Varianten: [Anpassen des senseBox:home Sketch](#)
3. Am einfachsten: [Wemos D1 mit senseBox Shield](#)
4. [Alle anderen ESP8266 Varianten](#)

Installation von **ESP8266 Arduino core**

Um den ESP8266 über die Arduino IDE programmierbar zu machen, wird eine passende Toolchain benötigt. Eine solche wurde als [ESP8266 Arduino core](#) von der ESP-Community entwickelt.

Zuallererst sollten die Softwarebibliotheken für die Sensoren installiert werden, falls dies noch nicht getan wurde (siehe [Anleitung](#)).

Danach muss die [ESP8266 Arduino core](#) Toolchain installiert werden. Diese Anleitung ist analog zur Anleitung aus dem [Github Repository](#):

1. Arduino starten
2. Im Menü unter `Datei -> Voreinstellungen` unter 'Additional Board Manager URLs' die Adresse
`http://arduino.esp8266.com/stable/package_esp8266com_index.json` einfügen
3. Im Menü `Werkzeuge -> Platine -> Boards Manager` kann man nun mit der Suche oben rechts im Fenster `ESP8266 by ESP8266 Community` installieren
4. Das entsprechende Board unter `Werkzeuge -> Platine` auswählen. Sollte nicht explizit das vorhandene Board zur Auswahl stehen, ist "Generic ESP8266" eine gute Wahl.

openSenseMap-Sketch für den ESP8266 anpassen

Damit der senseBox-Sketch auf dem ESP8266 läuft, sind ein paar Anpassungen nötig. Zuallererst benötigt man seinen senseBox-Sketch. Entweder man registriert eine neue senseBox ([Anleitung hier](#)), oder man verwendet seinen bestehenden Sketch. Wichtig ist dass man seine senseBox-ID und Sensor-IDs kennt.

Damit das Barometer (BMP280) korrekt in mit dem ESP8266 funktioniert, wird eine andere Bibliothek ([BME280](#)) zum ansteuern des Sensors benötigt. Diese kann [hier](#) heruntergeladen werden. Die Installation erfolgt analog zu unseren Bibliotheken, wie [hier](#) beschrieben ist.

Weiter [unten](#) findet ihr einen komplett angepassten Sketch. Dort müssen nur noch WiFi-Name und -Passwort, senseBox-ID und Sensor-IDs ersetzt werden.

Ethernet durch Wifi ersetzen

Da der ESP8266 über WiFi kommuniziert, brauchen wir die Ethernet Bibliothek nicht mehr. Dafür muss die ESP Wifi Bibliothek eingeladen und konfiguriert werden.

```
#include <Ethernet.h>
```

ersetzen durch

```
#include <ESP8266WiFi.h>
```

Die Zeilen welche die Ethernet Verbindung konfigurieren folgendermaßen ersetzen:

```
//Configure ethernet connection
IPAddress myIp(192, 168, 0, 42);
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
char server[] = "www.opensensemap.org";
EthernetClient client;
```

nach

```
//Configure wifi connection
char server[] = "www.opensensemap.org";
WiFiClient client;
const char* ssid      = "dein-wifi-name";
const char* password = "dein-wifi-passwort";
```

ändern.

Dann muss das Verbinden mit dem Ethernet mit der Wifi-Verbindung ersetzt werden.

Alt:

```
// start the Ethernet connection:
Serial.println("SenseBox Home software version 2.1");
Serial.println();
Serial.print("Starting ethernet connection...");
if (Ethernet.begin(mac) == 0) {
  Serial.println("Failed to configure Ethernet using DHCP");
  Ethernet.begin(mac, myIp);
} else Serial.println("done!");
```

Neu:

```
// start the Wifi connection:
Serial.println("SenseBox Home software version 2.1");
Serial.println();
Serial.print("Starting wifi connection...");
WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("done!");
```

Kompletter Sketch

```
/*
SenseBox Home - Citizen Sensingplatform
Version: 2.1
Date: 2016-07-23
Homepage: http://www.sensebox.de
Author: Jan Wirwahn & Gerald Pape, Institute for Geoinformatics, University of Muenster
Note: Sketch for senseBox Home Kit ESP8266
Email: support@sensebox.de
*/

#include <Wire.h>
#include "HDC100X.h"
#include "BMP280.h"
#include <Makerblog_TSL45315.h>
#include <SPI.h>
#include <ESP8266WiFi.h>

//SenseBox ID
#define SENSEBOX_ID "SENSEBOX_ID_AUS_DEM_ORIGINAL_SKETCH"

//Sensor IDs
#define TEMPSENSOR_ID "SENSOR_ID_AUS_DEM_ORIGINAL_SKETCH"
#define HUMISENSOR_ID "SENSOR_ID_AUS_DEM_ORIGINAL_SKETCH"
#define PRESSURESENSOR_ID "SENSOR_ID_AUS_DEM_ORIGINAL_SKETCH"
#define LUXSENSOR_ID "SENSOR_ID_AUS_DEM_ORIGINAL_SKETCH"
#define UVSENSOR_ID "SENSOR_ID_AUS_DEM_ORIGINAL_SKETCH"

//Configure wifi connection
char server[] = "www.opensensemap.org";
WiFiClient client;
const char* ssid      = "dein-wifi-name";
const char* password = "dein-wifi-passwort";

//Load sensors
Makerblog_TSL45315 TSL = Makerblog_TSL45315(TSL45315_TIME_M4);
HDC100X HDC(0x43);
BMP280 BMP;

//measurement variables
float temperature = 0;
float humidity = 0;
double tempBaro, pressure;
uint32_t lux;
uint16_t uv;
int messTyp;
#define UV_ADDR 0x38
#define IT_1 0x1

unsigned long oldTime = 0;
const unsigned long postingInterval = 60000;

void setup() {
    // Open serial communications and wait for port to open:
    Serial.begin(9600);
    // start the Wifi connection:
```

```

Serial.println("SenseBox Home software version 2.1");
Serial.println();
Serial.print("Starting wifi connection...");
WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("done!");
delay(1000);
//Initialize sensors
Serial.print("Initializing sensors...");
Wire.begin();
Wire.beginTransmission(UV_ADDR);
Wire.write((IT_1<<2) | 0x02);
Wire.endTransmission();
delay(500);
HDC.begin(HDC100X_TEMP_HUMI, HDC100X_14BIT, HDC100X_14BIT, DISABLE);
TSL.begin();
BMP.begin();
BMP.setOversampling(4);
Serial.println("done!");
Serial.println("Starting loop.");
temperature = HDC.getTemp();
}

void loop()
{
    // if there are incoming bytes available
    // from the server, read them and print them:
    if (client.available()) {
        char c = client.read();
        Serial.print(c);
        //Serial.write(c);
    }

    if (millis() - oldTime > postingInterval) {
        oldTime = millis();
        //----Pressure----//
        Serial.println("Posting pressure");
        messTyp = 2;
        char result = BMP.startMeasurment();
        if(result!=0){
            delay(result);
            result = BMP.getTemperatureAndPressure(tempBaro,pressure);
            postObservation(pressure, PRESSURESENSOR_ID, SENSEBOX_ID);
            //Serial.print("Temp_baro = ");Serial.println(tempBaro,2);
            //Serial.print("Pressure = ");Serial.println(pressure,2);
        }
        delay(2000);
        //----Humidity----//
        Serial.println("Posting humidity");
        messTyp = 2;
        humidity = HDC.getHumi();
        //Serial.print("Humidity = "); Serial.println(humidity);
        postObservation(humidity, HUMISENSOR_ID, SENSEBOX_ID);
        delay(2000);
        //----Temperature----//
        Serial.println("Posting temperature");
        messTyp = 2;
        temperature = HDC.getTemp();
        //Serial.println(temperature,2);
        //Serial.print("Temperature = ");Serial.println(temperature);
        postObservation(temperature, TEMPSENSOR_ID, SENSEBOX_ID);
        delay(2000);
        //----Lux----//
        Serial.println("Posting illuminance");
        messTyp = 1;
        lux = TSL.readLux();
        //Serial.print("Illumi = "); Serial.println(lux);
        postObservation(lux, LUXSENSOR_ID, SENSEBOX_ID);
    }
}

```

```

    delay(2000);
    //UV intensity
    messTyp = 1;
    uv = getUV();
    postObservation(uv, UVSENSOR_ID, SENSEBOX_ID);
}
}

void postObservation(float measurement, String sensorId, String boxId){
    char obs[10];
    if (messTyp == 1) dtostrf(measurement, 5, 0, obs);
    else if (messTyp == 2) dtostrf(measurement, 5, 2, obs);
    Serial.println(obs);
    //json must look like: {"value":"12.5"}
    //post observation to: http://opensensemap.org:8000/boxes/boxId/sensorId
    Serial.println("connecting...");
    String value = "{\"value\":";
    value += obs;
    value += "}";
    if (client.connect(server, 8000))
    {
        Serial.println("connected");
        // Make a HTTP Post request:
        client.print("POST /boxes/");
        client.print(boxId);
        client.print("/");
        client.print(sensorId);
        client.println(" HTTP/1.1");
        // Send the required header parameters
        client.println("Host:opensensemap.org");
        client.println("Content-Type: application/json");
        client.println("Connection: close");
        client.print("Content-Length: ");
        client.println(value.length());
        client.println();
        // Send the data
        client.print(value);
        client.println();
    }
    waitForResponse();
}

void waitForResponse()
{
    // if there are incoming bytes available
    // from the server, read them and print them:
    boolean repeat = true;
    do{
        if (client.available())
        {
            char c = client.read();
            Serial.print(c);
        }
        // if the servers disconnected, stop the client:
        if (!client.connected())
        {
            Serial.println();
            Serial.println("disconnecting.");
            client.stop();
            repeat = false;
        }
    }
    while (repeat);
}

uint16_t getUV(){
    byte msb=0, lsb=0;
    uint16_t uvValue;

    Wire.requestFrom(UV_ADDR+1, 1); //MSB
    delay(1);
    if(Wire.available()) msb = Wire.read();
}

```

```
Wire.requestFrom(UV_ADDR+0, 1); //LSB
delay(1);
if(Wire.available()) lsb = Wire.read();

uvValue = (msb<<8) | lsb;

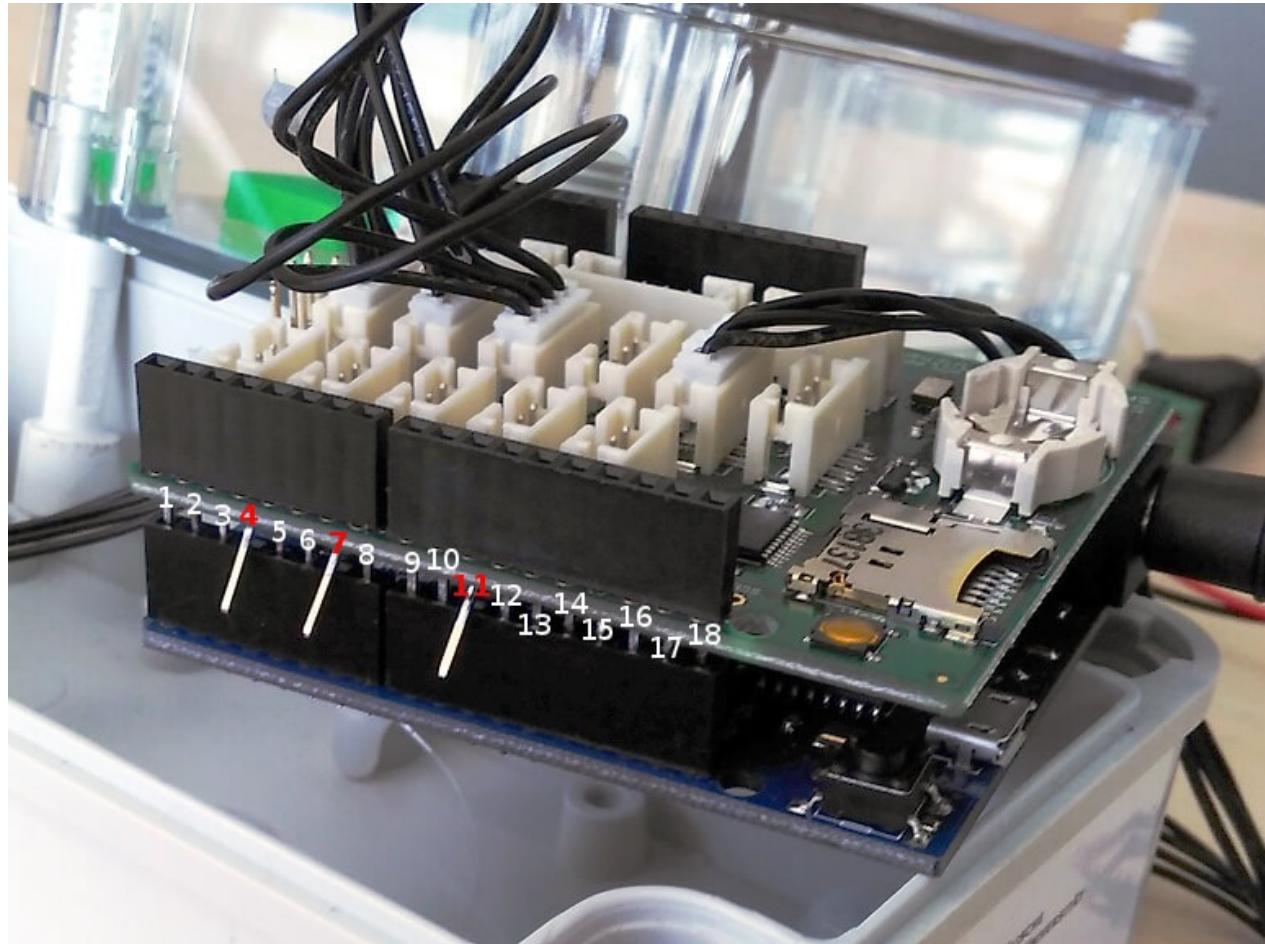
return uvValue*5;
}
```

senseBox:home mit Wemos D1 und senseBox Shield

Der [Wemos D1](#) ist ein Arduino Uno kompatibles Board mit einem ESP8266 als Mikrokontroller.

Anpassungen am senseBox Shield

Im Prinzip ist das senseBox:home Shield kompatibel mit dem Wemos D1. Damit das senseBox Shield mit dem Wemos D1 zusammenarbeitet, müssen ein paar der Pins abgebogen werden. Auf diesem Bild siehst du welche:



Achtung: Für defekte Shields oder Sensoren übernehmen wir keine Haftung!

senseBox:home mit anderen ESP8266 Varianten

Wie Eingangs schon erwähnt, ist der ESP8266 Mikrokontroller weitestgehend mit dem Arduino kompatibel. Die senseBox verwendet den I²C Bus um mit den Sensoren zu kommunizieren. Dieser verwendet standardmäßig Pin 4 (SDA) und 5 (SCL).

ESP8266-01

Das Modell 01 des ESP8266 hat nur Pin 0 und 2 auf der Hardware verfügbar. Hier muss zum initialisieren des I²C Bus `Wire.begin(0, 2)` verwendet werden.

Hinweise zum Debuggen

- Leuchtet die Blaue LED auf, kommen Daten auf dem RX Pin an
- Beim Booten gibt der Bootloader des ESP Debuginformationen mit 74880 Baud aus, anschließend läuft NodeMCU (sofern installiert) mit 115200 Baud.

Der ESP hat verschiedene Bootmodi, welche über die Spannung an GPIO 0 und 2 gewählt werden. Um den ESP normal zu booten, sodass dieser sein Programm ausführt, benötigen beide Pins einen Pullup auf 3.3V. Um ein Programm auf den ESP zu schreiben muss der UART-Modus verwendet werden, welcher mit einem Pulldown auf GPIO 0 und einem Pullup auf GPIO 2 aktiviert wird.

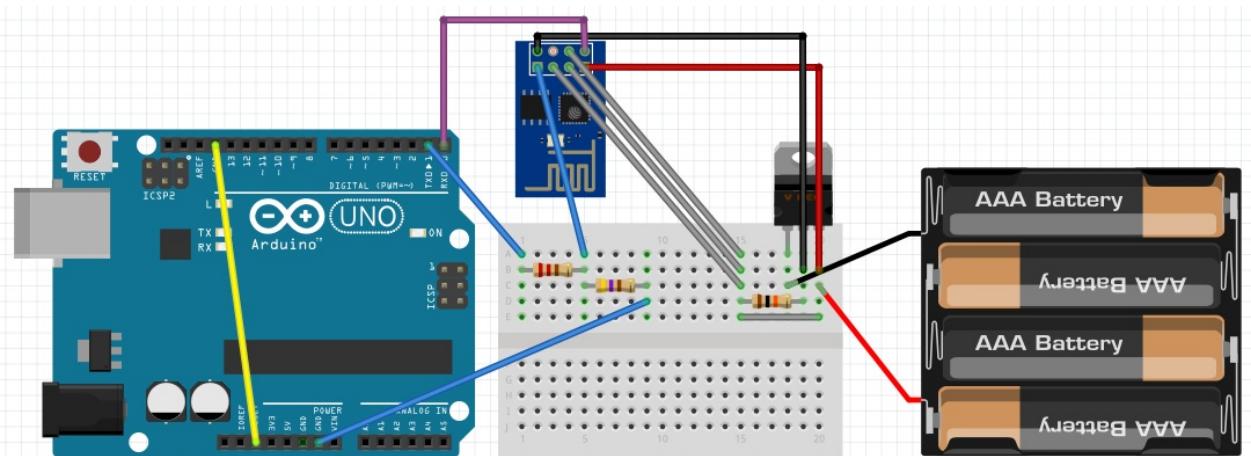
- Siehe [hier](#) für Informationen zu Pullup-Widerständen.
- Siehe [hier](#) für Informationen zu den Bootmodi des ESP8266.

TTL-bridge über Arduino

Die kompakteren ESP Modelle haben meist keinen USB Anschluss über welchen der serielle Port des Mikrocontrollers angesprochen werden könnte. Um solche Boards zu programmieren ist eine separater USB-TTL-converter notwendig. Falls gerade keine dedizierte Hardware vorliegt, kann ein Arduino dazu zweckentremdet werden!

Legt man den `RST` Pin des Arduinos auf `GND`, wird effektiv der Prozessor des Arduinos deaktiviert. Dadurch geben die Pins 0 und 1 die Signale des seriellen Busses unverändert weiter.

Da Arduinos meist mit 5V und ESPs mit 3,3V arbeiten, muss die Spannung der Signale auf ein einheitliches Level gebracht werden: Am einfachsten wird dazu der `rx`-Output des Arduinos durch einen Voltage-Divider auf 3,3V gebracht (blaue Kabel). Die `tx` Leitung (lila Kabel) kann direkt verbunden werden, da der Arduino auch 3,3V-Spannungen als `HIGH` versteht. Die vollständige Schaltung - inklusive Pullups (graue Kabel) und Stromversorgung des ESP8266 - sieht also wie folgt aus:



Hinweis: Der `3.3v` -Pin des Arduinos stellt nicht genügend Strom bereit, um den ESP8266 zu betreiben. Hierzu sollte eine separate Stromversorgung (3,3V!) verwendet werden! Siehe [hier](#) für Informationen zu Voltage Dividern.

Nun kann der Rechner mit dem ESP direkt kommunizieren. Zum Testen einfach den seriellen Monitor der Arduino IDE öffnen, und Kommandos an den ESP schicken!