

# **Umweltdatenmessung mit dem Raspberry Pi**

Vorwissenschaftliche Arbeit verfasst von

**Lukas Winkler**

Klasse 8A



Betreuer: MMag. Matthias Kittel

BRG Rechte Kremszeile  
Rechte Kremszeile 54  
3500 Krems an der Donau

Krems an der Donau, Januar 2015

Diese Arbeit wurde mit Texmaker geschrieben, in Palatino mit Hilfe von [pdfLATEX](#) und [Biber](#) gesetzt.

Die LATEX Vorlage von Karl Voit basiert auf [KOMA script](#) und steht im Internet zum Download bereit: <https://github.com/novoid/LaTeX-KOMA-template>

# Abstract

This is a placeholder for the abstract. It summarizes the whole thesis to give a very short overview. Usually, this the abstract is written when the whole thesis text is finished.

# Inhaltsverzeichnis

<b>Abstract</b>	<b>iii</b>
<b>1. Einleitung</b>	<b>1</b>
<b>2. Hardware</b>	<b>3</b>
2.1. Der Raspberry Pi . . . . .	3
2.1.1. Geschichte . . . . .	4
2.1.2. Technische Daten . . . . .	4
2.2. Sensoren . . . . .	4
2.2.1. Temperatur . . . . .	5
2.2.2. Luftfeuchtigkeit . . . . .	6
2.2.3. Luftdruck . . . . .	6
2.2.4. Luftqualität . . . . .	7
2.3. Display . . . . .	8
2.4. Anschluss . . . . .	9
<b>3. Software</b>	<b>13</b>
3.1. main.sh . . . . .	13
3.1.1. Allgemeines . . . . .	13
3.1.2. Messung . . . . .	14
3.1.3. Speichern, Aufbereiten und Verarbeiten . . . . .	16
3.2. Display . . . . .	18
3.3. Webinterface . . . . .	18
3.3.1. Livedaten . . . . .	19
3.3.2. Diagramme . . . . .	21
3.4. Endauswertung . . . . .	22
3.5. diverses . . . . .	25
3.5.1. Weather Underground . . . . .	25
3.5.2. Autostart . . . . .	25
3.5.3. mitternacht.sh . . . . .	25
3.5.4. sonstiges . . . . .	26

## Inhaltsverzeichnis

<b>4. Auswertung</b>	<b>27</b>
4.1. Aufzeichnung . . . . .	27
4.2. Graphische Darstellung . . . . .	27
4.3. Endauswertung . . . . .	31
<b>5. Fazit</b>	<b>32</b>
5.1. Ausblick . . . . .	33
<b>A. Weitere Informationen</b>	<b>34</b>
<b>B. Präsentationen</b>	<b>35</b>
<b>Literatur</b>	<b>38</b>
<b>Abbildungsverzeichnis</b>	<b>41</b>
<b>Dateiverzeichnis</b>	<b>43</b>
<b>Glossar</b>	<b>44</b>

# 1. Einleitung

Im letzten Jahr habe ich mich damit beschäftigt, wie man mithilfe eines Raspberry Pi Umweltdaten messen, aufzeichnen und auswerten kann. Hierzu verwende ich mehrere Sensoren, die die Lufttemperatur (sowohl im Klassenraum, als auch außen), Luftfeuchtigkeit, Luftdruck und die relative Luftqualität. Diese Daten werden als [CSV-Datei](#) gespeichert und können grafisch und rechnerisch ausgewertet werden.

Die grafische Auswertung läuft über ein Webinterface, das innerhalb der Schule aufrufbar ist. Von außerhalb ist eine regelmäßig aktualisierte Kopie unter [winkler.kremszeile.at](http://winkler.kremszeile.at) erreichbar. Auf dieser Seite können neben allgemeinen Informationen über das Projekt und Links zu weiteren Informationen<sup>1</sup> die aktuellen Messwerte als Balken-Diagramm, welches sich selbst aktualisiert, und die komplette Aufzeichnung als interaktives Diagramm dargestellt werden.

Unabhängig davon können mit einem von mir geschriebenen [Python](#)-Programm die Daten einer Messung im Nachhinein mathematisch ausgewertet werden.

Um diese Vorwissenschaftliche Arbeit so verständlich wie möglich zu halten, werden Wörter, die im Text farbig hervorgehoben sind, im Glossar auf Seite [44](#) erklärt.

---

<sup>1</sup>siehe Anhang A

## 1. Einleitung



Abbildung 1.1.: Messstation

## 2. Hardware

Bevor man Daten auswerten kann, muss man diese erst aufzeichnen. Hierzu wird ein *Raspberry Pi* verwendet, welcher die komplette Aufzeichnung und Auswertung steuert. An diesem sind sämtliche Sensoren und das Display zur Anzeige der aktuellen Messwerte angeschlossen.

### 2.1. Der Raspberry Pi

Der *Raspberry Pi* ist ein *Einplatinencomputer*, der 2012 von der *Raspberry Pi Foundation* auf den Markt gebracht wurde.

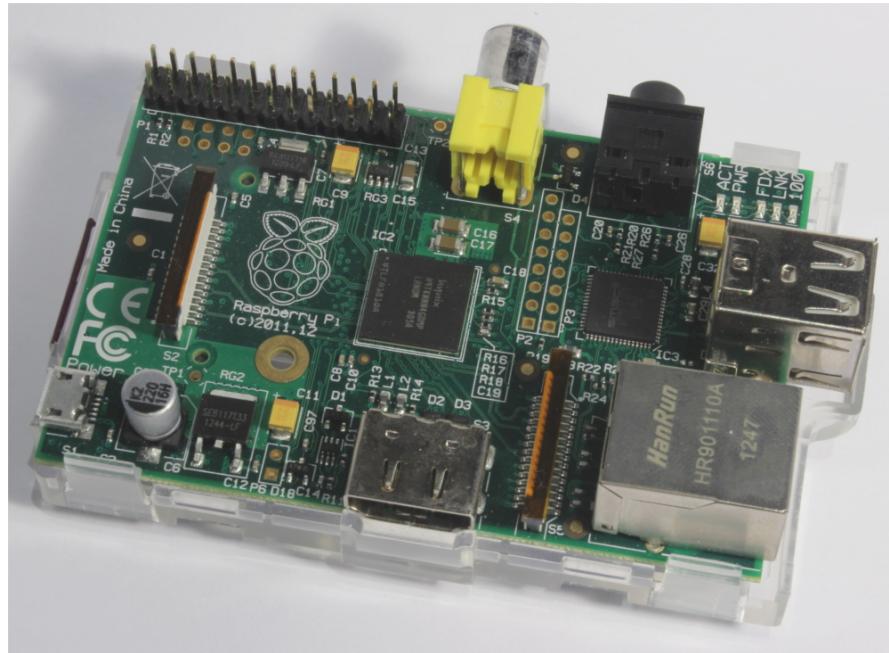


Abbildung 2.1.: Raspberry Pi - Modell B

## 2. Hardware

### 2.1.1. Geschichte

Ursprünglich war der Raspberry Pi als günstiger Computer gedacht, um britischen Jugendlichen das Programmieren näher zu bringen. An der *University of Cambridge* stellte man fest, dass die Vorkenntnisse von Studienanfängern immer geringer wurden, weil sie – sowohl privat als auch in der Schule – sich immer weniger mit der Funktionsweise von Computern und Programmen beschäftigen. Daher wollte man einen Computer entwickeln, mit dem die Jugendlichen experimentieren können.<sup>2,3</sup>

Inzwischen wurden 3,8 Millionen Stück verkauft (Stand Oktober 2014<sup>4</sup>) und 6 verschiedene Modelle entwickelt. (Bis Februar 2015)

### 2.1.2. Technische Daten

Die Technik in einem Raspberry Pi ist vergleichbar mit der eines Smartphones. Der Raspberry Pi hat eine **CPU** mit **700 MHz**, welche auf bis zu **1 GHz** übertaktbar ist, und je nach Modell 256 oder 512 MB Arbeitsspeicher. Als Speicher für das Betriebssystem (verschiedene **Linux-Distributionen** stehen zur Auswahl) wird eine SD-Karte bzw. eine microSD-Karte verwendet.

Zur Stromversorgung genügt ein normales Handyladegerät mit Micro-USB-Anschluss und mindestens **1 Ampere** Stromstärke, denn der Raspberry Pi verbraucht nur 3.5 Watt<sup>5</sup> (Modell B).

Zum Anschließen anderer Hardware gibt es zwei USB-Anschlüsse und **26 GPIO-Pins**.

## 2.2. Sensoren

Zur Messung der Umweltdaten werden folgende Sensoren verwendet:

- 4 Temperatursensoren **DS18B20** ([2.2.1](#))

---

<sup>2</sup>Raspberry Pi Foundation, *The Making of Pi*.

<sup>3</sup>Wikipedia, *Raspberry Pi— Wikipedia, Die freie Enzyklopädie*, Geschichte.

<sup>4</sup>@Raspberry\_Pi, "@ruskin147 As of today, it looks like 3.8 million - that's an \*awful lot of computers\*."

<sup>5</sup>elinux, *RPi Hardware - Power*.

## 2. Hardware

- Luftfeuchtesensor *DHT22* ([2.2.2](#))
- Luftdrucksensor *BMP085* ([2.2.3](#))
- Luftqualitätssensor *VOLTCRAFT CO-20* ([2.2.4](#))
- *CPU*-Temperatur des Raspberry Pi

### 2.2.1. Temperatur

Mithilfe von 4 Sensoren des Typs *DS18B20* werden die Innentemperatur, die Gehäusetemperatur und die Bodentemperatur (Außen) gemessen. Diese haben eine Messgenauigkeit von  $\pm 0.5 \text{ }^{\circ}\text{C}$  und einen Messbereich von  $-10 \text{ }^{\circ}\text{C}$  bis  $85 \text{ }^{\circ}\text{C}$ .<sup>6</sup>

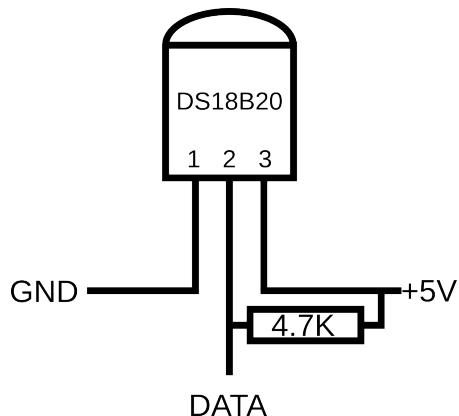


Abbildung 2.2.: Pinbelegung des DS18B20

Der Sensor wird mithilfe von einem [1-Wire-Datenbus](#) ausgelesen. Hierbei benötigt man (außer für die Stromversorgung mit 5 Volt nur ein Kabel, auf dem die Daten übertragen werden.<sup>7</sup> Zusätzlich wird ein  $4.7\text{k}\Omega$  Widerstand zwischen dem Pin für Daten und dem Pin für +5 V benötigt. (siehe Abbildung 2.2) Ein weiterer Vorteil von 1-Wire ist, dass nahezu beliebig viele Sensoren auf einem Datenkabel parallel geschaltet werden können.

Die Messdaten des *DS18B20* können auf dem Raspberry Pi sehr einfach ausgelesen werden, weil dies von einem Linux-[Kernelmodul](#) erledigt wird. Um die Temperatur zu erhalten, muss nur eine [Gerätedatei](#) ausgelesen werden,

<sup>6</sup>Maxim Integrated Products, [DS18B20 - Data Sheet](#), S. 20.

<sup>7</sup>FHEMWiki, [Kategorie:1-Wire - FHEMWiki](#).

## 2. Hardware

welche das Messergebnis in tausendstel Grad Celsius enthält. (Siehe Abbildung 2.3)

```
pi@raspberry /sys/bus/w1/devices $ cat 10-00080277a5db/w1_slave 10-00080277abe1/w1_slave
2f 00 4b 46 ff ff 08 10 78 : crc=78 YES
2f 00 4b 46 ff ff 08 10 78 t=23250
2f 00 4b 46 ff ff 08 10 78 : crc=78 YES
2f 00 4b 46 ff ff 08 10 78 t=23250
pi@raspberry /sys/bus/w1/devices $ cat 10-00080277a5db/w1_slave 10-00080277abe1/w1_slave
2f 00 4b 46 ff ff 01 10 ca : crc=ca YES
2f 00 4b 46 ff ff 01 10 ca t=23687
2f 00 4b 46 ff ff 07 10 60 : crc=60 YES
2f 00 4b 46 ff ff 07 10 60 t=23312
```

Abbildung 2.3.: Die erste erfolgreiche Messung

### 2.2.2. Luftfeuchtigkeit

Zum Messen der Luftfeuchtigkeit der Außenluft wird der *DHT22* verwendet. Dieser kann auch die Temperatur messen. Die Messgenauigkeit beträgt  $\pm 0.5\text{ }^{\circ}\text{C}$  und  $\pm 2\%$  relative Luftfeuchte.<sup>8</sup> Wie der *DS18B20* (2.2.1) benötigt der Luftfeuchtigkeitssensor zusätzlich zur Stromversorgung nur ein Kabel zur Datenübertragung. Es können jedoch nicht mehrere Sensoren parallel geschaltet werden.<sup>9</sup>

Die Daten des Sensors werden von einem C-Programm von *Adafruit* ausgelesen.<sup>10</sup>

### 2.2.3. Luftdruck

Der *BMP085* ist der präziseste Sensor. Er wird zum Messen des Luftdruckes und der Außentemperatur verwendet und hat dabei eine Genauigkeit von  $\pm 1.0\text{ hPa}$  und  $0.5\text{ }^{\circ}\text{C}$  bei  $25\text{ }^{\circ}\text{C}$ <sup>11</sup>

Die Messdaten überträgt der Sensor über einen **I<sup>2</sup>C-Bus**. Dabei werden (zusätzlich zur Stromversorgung) **zwei** Kabel zur Datenübertragung benötigt.

---

<sup>8</sup>Aosong Electronics Co.,Ltd, *Digital-output relative humidity & temperature sensor/module DHT22*.

<sup>9</sup>Adafruit User LADY ADA, *DHT Humidity Sensing on Raspberry Pi or Beaglebone Black with GDocs Logging*, Wiring.

<sup>10</sup>Ebd., Software Install.

<sup>11</sup>Bosch Sensortec, *BMP085 Digital pressure Sensor - Data Sheet*, S. 6.

## 2. Hardware

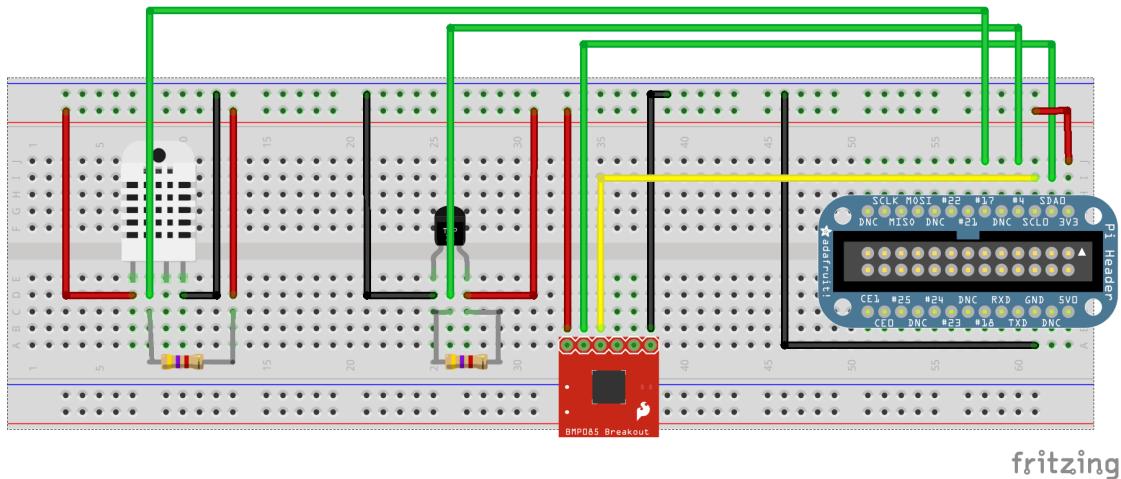


Abbildung 2.4.: Anschlusskizze von DS18B20 (Mitte; 2.2.1), DHT22 (Links; 2.2.2) und BMP085 (Rechts; 2.2.3)

Über eines (in Abbildung 2.4 gelb) schickt der Raspberry Pi dem Sensor die Taktfrequenz, in der er die Daten übertragen soll, und im anderen (grün) werden die eigentlichen Daten übertragen.<sup>12</sup>

Auch hier werden die Daten von einem Programm von Adafruit ausgelesen.<sup>13</sup>

### 2.2.4. Luftqualität

Der letzte Sensor, der hinzugekommen ist, ist der VOLTCRAFT CO-20. Da CO<sub>2</sub>-Sensoren und andere genaue Luftqualitätssensoren teuer sind, habe ich mich für einen einfachen VOC-Sensor entschieden. Dieser misst die Menge an *Flüchtigen organischen Verbindungen* in der Luft. Dies sind Stoffe, die schon bei niedrigen Temperaturen verdampfen. Sie können von verschiedenen Quellen stammen (z. B.: Benzindämpfe, Tabakrauch, Lacke)<sup>14</sup> und von leichten Kopfschmerzen und Konzentrationsstörungen bis zu bleibenden Gesundheitsschäden führen.<sup>15</sup>

<sup>12</sup>Adafruit User KEVIN TOWNSEND, [Using the BMP085/180 with Raspberry Pi or Beaglebone Black](#), Hooking Everything Up.

<sup>13</sup>Ebd., Using the Adafruit BMP Python Library (Updated).

<sup>14</sup>Innenraumlufthygiene-Kommission des Umweltbundesamtes, [LEITFADEN FÜR DIE INNENRAUMHYGIENE IN SCHULGEBÄUDEN](#), S. 41 ff.

<sup>15</sup>WISSEN Wiki, [Flüchtige organische Verbindung](#), Gesundheitliche Wirkung.

## 2. Hardware

Der Sensor gibt einen Wert an, der die relative Verschlechterung seit dem Einschalten angibt. Hierbei steht 450 für die anfängliche Qualität ist und ein höherer Wert für eine schlechtere Luftqualität. Da der *VOLTCRAFT CO-20* jedoch nicht mehr erhältlich ist, verwende ich den baugleichen *Raumluftfühler* von Velux.<sup>16</sup>



Abbildung 2.5.: Velux Raumluftfühler

Der Sensor wird über USB an den Raspberry Pi angeschlossen. Um die Daten unter Linux auszulesen, wird das Programm *usb-sensors-linux*<sup>17</sup> verwendet.

## 2.3. Display

Damit nicht immer ein Computer benötigt wird, um die aktuellen Messwerte zu erfahren, verwende ich ein Display, welches diese anzeigt. Ursprünglich habe ich ein 16x2 Zeichen *LC-Display* von *Conrad Electronic* verwendet.<sup>18</sup> Dieses wird nach der Anleitung von *schnatterente.net*<sup>19</sup> angeschlossen.

---

<sup>16</sup>Velux, [VELUX Raumluftfühler](#).

<sup>17</sup>usb-sensors-linux, [Install AirSensor on Linux](#).

<sup>18</sup>ANAG VISION, [AV1624 Datasheet](#).

<sup>19</sup>Schnatterente.net, [Raspberry Pi: 32 Zeichen Hitachi HD44780 Display](#).

## 2. Hardware

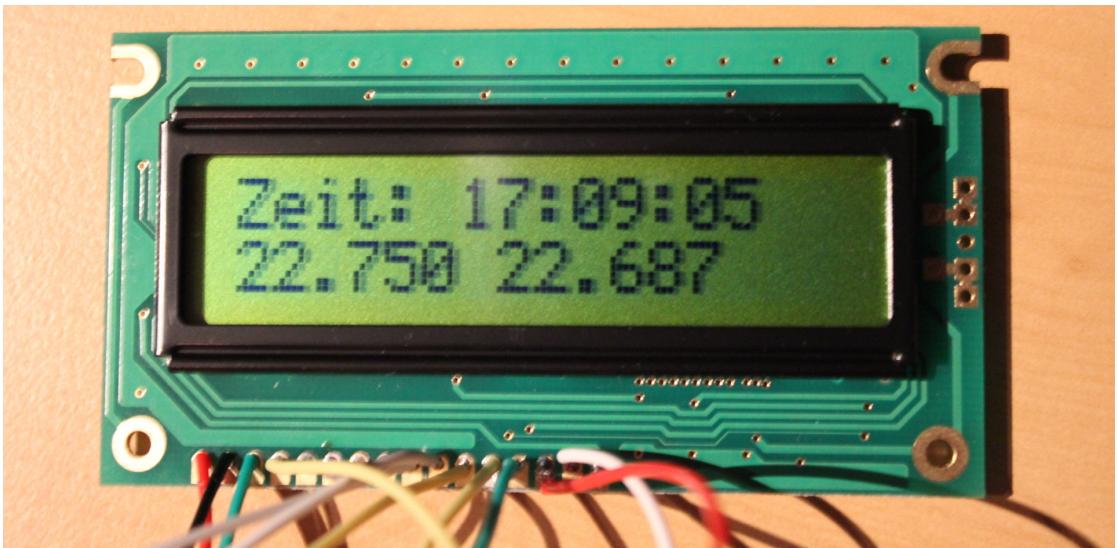


Abbildung 2.6.: Das erste Display

Da jedoch fix angelötete Kabel unflexibel sind, bin ich auf ein Display von [Pollin.de<sup>20</sup>](#) umgestiegen, welches mit einer Steckverbindung angeschlossen wird.

### 2.4. Anschluss

Anfangs wurde sämtliche Hardware auf einem [Steckbrett](#) aufgesteckt und mit einzelnen Kabeln verbunden. Dies ist sehr praktisch für schnelle Versuche und häufige Änderungen, ist aber sehr unstabil und nicht transportabel.

Daher wurden alle Sensoren auf [Streifenplatinen](#) gelötet, welche mit Steckverbindungen und Kabeln miteinander verbunden werden.

Die komplette Hardware kommt nun in eine Holzbox, mit einer Öffnung für das Display und 3 [LEDs](#) auf der Vorderseite. Eine grüne [LED](#) signalisiert, dass die Aufzeichnung gerade läuft. Eine gelbe [LED](#) leuchtet kurz auf, wenn gerade eine Messung aufgrund eines Fehlers wiederholt wird. Und eine rote [LED](#) zeigt, dass die Aufzeichnung gerade gestoppt ist. Zusätzlich gibt es am rechten Rand eine Öffnung für die USB-Anschlüsse und zwei

---

<sup>20</sup>Pollin.de, [LCD-Modul TC1602E-01](#).

## 2. Hardware

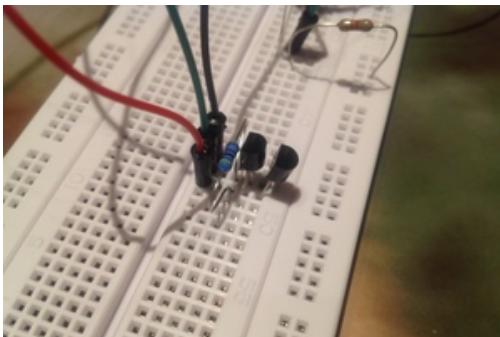


Abbildung 2.7.: DS18B20 (2.2.1)

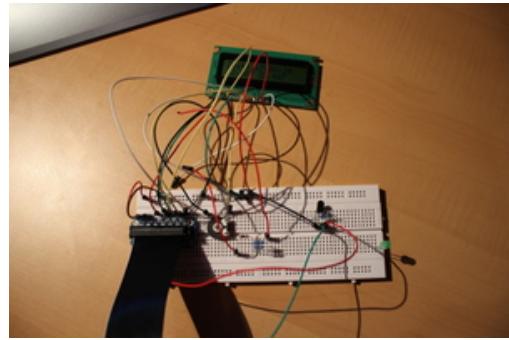


Abbildung 2.8.: Steckbrett mit Display (2.3)

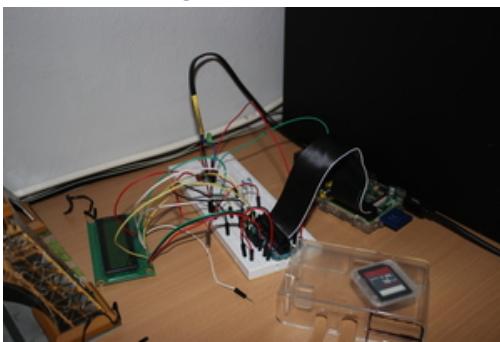


Abbildung 2.9.: erste  
mehrwöchige  
Messung

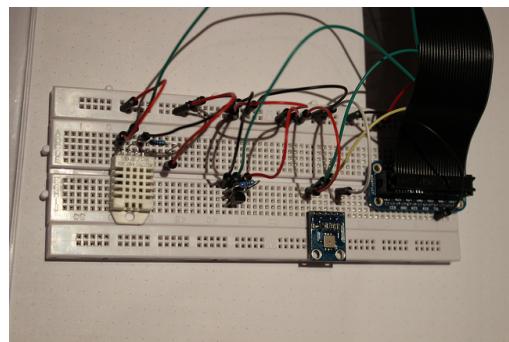


Abbildung 2.10.: neue Sensoren auf dem  
Steckbrett

Steckverbindungen für die Außensensoren. Am linken Rand ist eine kleiner Spalt für die Stromversorgung.

## 2. Hardware

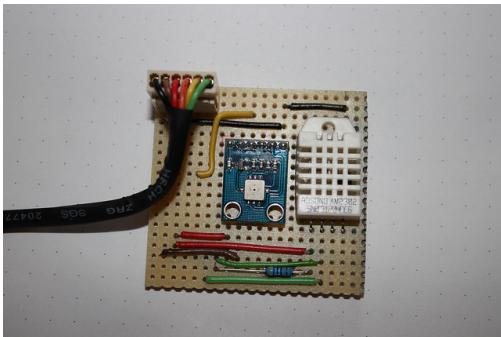


Abbildung 2.11.: Außensensoren

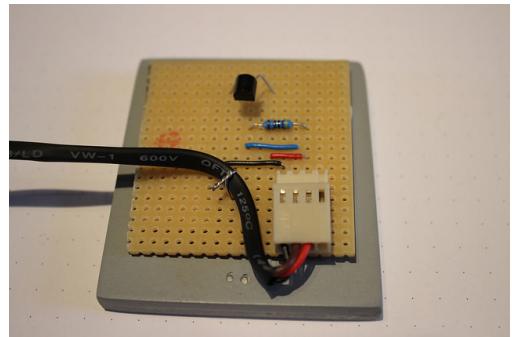


Abbildung 2.12.: Innensensoren

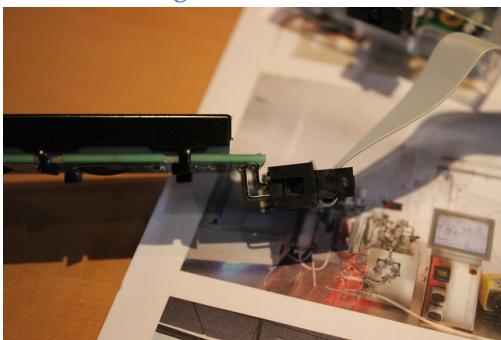


Abbildung 2.13.: Steckverbindung am Display

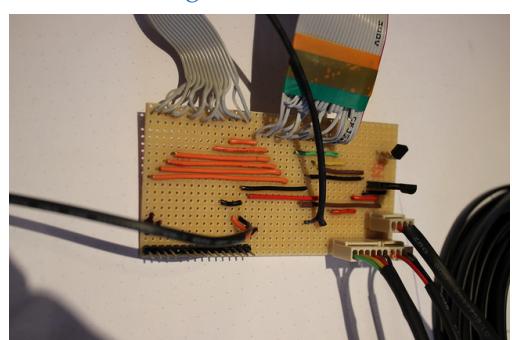


Abbildung 2.14.: Hauptplatine

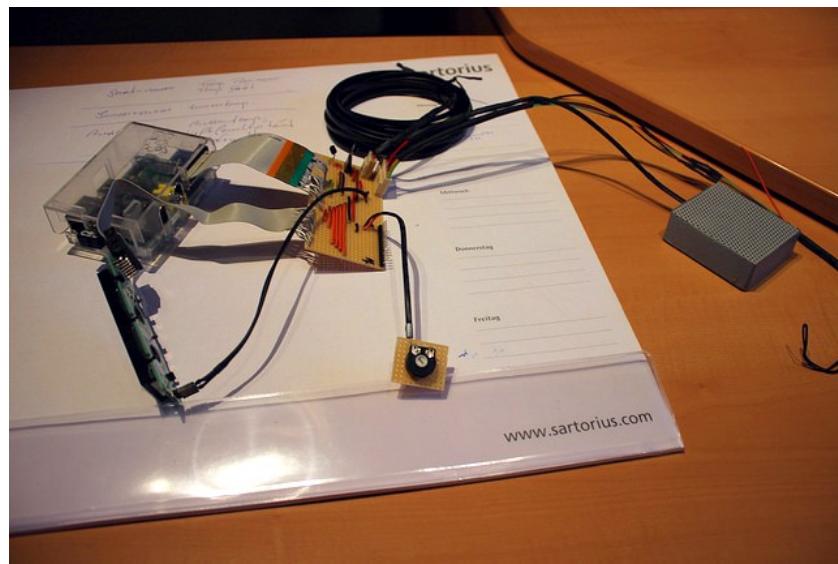


Abbildung 2.15.: Komplette Hardware

## 2. Hardware

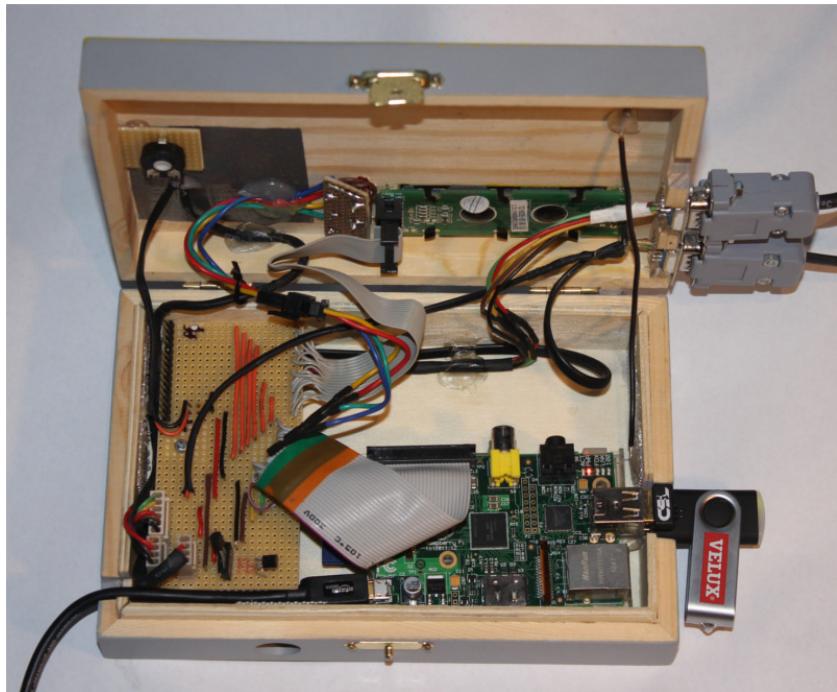


Abbildung 2.16.: Die Hardware in der Holzbox

# 3. Software

Die Software, die verwendet wird, teilt sich in 5 Teile auf:

- Auslesen der Sensoren, Aufbereiten der Daten und allgemeine Steuerung (*main.sh*)
- Steuern des Displays
- Webinterface (Livedaten, grafische Darstellung und Informationen)
- mathematische Endauswertung
- sonstiges

## 3.1. main.sh

Das wichtigste Programm ist das **Bash**-Script *main.sh*. Mithilfe eines solchen können Programme gesteuert und ihre Ausgaben verarbeitet werden. Dieses Script kümmert sich um die Aufzeichnung, erste Verarbeitung und Speicherung der Daten und die Steuerung der anderen Programme.

### 3.1.1. Allgemeines

Zunächst werden die Pins angegeben, an denen die **LEDs** angeschlossen sind. In den Zeilen 11-13 wird nun die grüne **LED** eingeschaltet, um zu zeigen, dass das Programm läuft.

```
8 gpio mode 13 out # gelb
9 gpio mode 12 out # rot
10 gpio mode 3 out #grün
11 gpio write 13 0 # nur grün einschalten
12 gpio write 12 0
13 gpio write 3 1
```

### 3. Software

Datei 1: main.sh (Zeile 8 bis 13)

Nun startet das eigentliche Programm. Alles, was nun folgt wird wiederholt, bis die Aufzeichnung beendet wird.

```
27 while true  
28 do
```

Datei 2: main.sh (Zeile 27 bis 28)

In den folgenden drei Zeilen wird der aktuelle Zeitpunkt in drei verschiedenen Formaten für drei verschiedene Zwecke gespeichert.

Code	Beispiel	Verwendung
%Y/%m/%d %H:%M:%S	2014/11/23 16:47:50	Format zum Abspeichern in <a href="#">CSV-Datei</a>
%d.%m %H:%M:%S	23.11 16:47:50	einfach lesbares Format für das Display
%d.%m.%y %H:%M:%S	23.11.2014 16:47:50	einfaches, exaktes Format für das Webinterface

Tabelle 3.1.: Datumsformate

```
29 uhrzeit=$( date +%Y/%m/%d\ %H:%M:%S )  
30 uhrzeit_display=$( date +%d.%m\ %H:%M:%S )  
31 uhrzeit_lang=$( date +%d.%m.%y\ %H:%M:%S )
```

Datei 3: main.sh (Zeile 29 bis 31)

#### 3.1.2. Messung

Als erstes werden die Sensoren ausgelesen. Am einfachsten kann mit dem im Raspberry Pi integrierten Thermometer die [CPU](#)-Temperatur ausgelesen werden. Hierzu müssen nur die Zeichen 6-9 (weil die gesamte Ausgabe *temp=45.5'C* lautet) aus einer [Gerätedatei](#) gelesen werden:

### 3. Software

```
32 rasp=$( /opt/vc/bin/vcgencmd measure_temp | cut -c 6,7,8,9 )
```

Datei 4: main.sh (Zeile 32)

Nur wenig aufwändiger ist das Auslesen der Temperatursensoren (*DS18B20*, siehe 2.2.1). Auch hier wird eine **Gerätedatei** ausgelesen, jedoch ist hier die Ausgabe umfangreicher. (siehe Abbildung 2.3 auf Seite 6) Daher muss zunächst mit *grep* und *awk* die Zahl herausgeschnitten werden. Um nun von tausendstel Grad Celsius zu °C zu kommen, wird mithilfe des *basic calculator* (*bc*) durch 1000 dividiert.

Da die Sensoren manchmal ungültige Werte zurückgeben, wird nach der ersten Messung überprüft, ob dies der Fall ist (Zeile 34) und die Messung solange wiederholt, bis eine gültige Messung erfolgt.

```
33 temp1=$( echo "scale=3; $(grep 't=' /sys/bus/w1/devices/
    w1_bus_master1/10-000802b53835/w1_slave | awk 'F[1] ~
    '/{print $2}')/1000" | bc -l )
34 while [ "$temp1" == "-1.250" ] || [ "$temp1" == "85.000" ]
35 do
36     gpiowrite 13 1
37     echo "Temp1: $temp1"
38     temp1=$( echo "scale=3; $(grep 't=' /sys/bus/w1/devices/
        w1_bus_master1/10-00080277abe1/w1_slave | awk 'F[1] ~
        '/{print $2}')/1000" | bc -l )
39     gpiowrite 13 0
40 done
```

Datei 5: main.sh (Zeile 33 bis 40)

Die Adafruit-Programme<sup>2122</sup>, die den Luftfeuchtesensor (siehe 2.2.2) und den Luftdrucksensor (siehe 2.2.3) auslesen, geben die Feuchtigkeit und die Temperatur durch einen Strichpunkt getrennt an. Daher wird dies zu Beginn als Trennzeichen angegeben.

<sup>21</sup>Adafruit User LADY ADA, *DHT Humidity Sensing on Raspberry Pi or Beaglebone Black with GDocs Logging*.

<sup>22</sup>Adafruit User KEVIN TOWNSEND, *Using the BMP085/180 with Raspberry Pi or Beaglebone Black*.

### 3. Software

```
4 IFS=";_"
```

Datei 6: main.sh (Zeile 4)

Dadurch kann die Ausgabe einfach aufgetrennt werden:

```
66 luft_roh=$(sudo python /home/pi/Temperaturmessung/
    Fremddateien/AdafruitDHT.py 2302 17)
67 set -- $luft_roh
68 luft_temp=$1
69 luft_feucht=$2
```

Datei 7: main.sh (Zeile 66 bis 69)

Auch hier wird bei ungültigen Messwerten mehrmals gemessen.

Der relative Wert für die Luftqualität wird direkt von *usb-sensors-linux*<sup>23</sup> zurückgegeben und muss daher nicht weiterbearbeitet werden.

```
84 qualitat=$(sudo /home/pi/Temperaturmessung/Fremddateien/
    airsensor -v -o)
```

Datei 8: main.sh (Zeile 84)

#### 3.1.3. Speichern, Aufbereiten und Verarbeiten

Nachdem alle Sensoren ausgelesen wurden, müssen sie dauerhaft gespeichert werden. Hierzu werden alle Werte durch ein Komma getrennt und als neue Zeile an die bisherigen Messungen angehängt.

```
89 ausgabe=${uhrzeit}\,$temp1\,$temp2\,$temp3\,$temp4\,
    ${luft_temp}\,$luft_feucht\,$druck\,$temp_druck\,
    ${rasp}\,$qualitat
90 echo $ausgabe >>/home/pi/Temperaturmessung/dygraphs.csv
```

Datei 9: main.sh (Zeile 89 bis 90)

Hierdurch entsteht eine **CSV-Datei**-Datei die wie folgt aussehen kann:

<sup>23</sup>usb-sensors-linux, *Install AirSensor on Linux*.

### 3. Software

```
1 2014/10/03 12:47:36,27.562,29.437,17.375,29.437,19.1,71.4,1000.95,19.30,53.0,1181
2 2014/10/03 12:48:07,27.625,29.437,17.375,29.437,19.1,71.4,1000.86,19.30,53.0,1140
3 2014/10/03 12:48:34,27.625,29.437,17.437,29.500,19.2,71.5,1001.00,19.40,53.0,1151
4 2014/10/03 12:49:02,27.625,29.500,17.437,29.500,19.2,71.5,1000.85,19.40,53.0,1147
```

Datei 10: dygraphs.csv

Diese Datei wird in den Ordner des Webservers kopiert, damit sie grafisch dargestellt werden kann (siehe [3.3.2](#)). Weiters verwendet die *Endauswertung* (siehe [3.4](#)) auch diese Datei zur rechnerischen Auswertung.

Als nächstes wird der Text für das Display (siehe [2.3](#)) erzeugt. Da dort der Platz beschränkt ist (16x2 Zeichen), werden alle Messwerte um 3 Stellen (bzw. 2 bei Luftdruck) gekürzt. Anschließend werden diese Daten in *text.txt* (für Display) und *text\_ws.txt* (für Webinterface) exportiert.

```
92 temp1_r=$( echo $temp1 | rev | cut -c 3- | rev )
```

Datei 11: main.sh (Zeile 92)

```
1 03.10 14:49:02
2 27.3
3 30.6
4 25.7
5 30.6
6 25
7 50
8 25.7
9 989.5
10 59.5
11 450
```

Datei 12: text.txt

```
1 03.10.14 14:49:02,27.3,30.6,25.7,30.6,25,50,25.7,989.5,59.5,450
```

Datei 13: text\_ws.txt

Abschließend wird noch 8 Sekunden gewartet und jedes tausende Mal ein Backup gemacht und mir per E-Mail geschickt, bevor die nächste Messung von vorne beginnt.

### 3. Software

## 3.2. Display

Um die aktuellen Messungen auch ohne Computer zu sehen, werden sie auch direkt am Raspberry Pi auf einem Display angezeigt (siehe auch 2.3). Um das Display anzusteuern wird ein Programm<sup>24</sup> von [schnatterente.net](http://schnatterente.net) verwendet. Dieses wurde von mir um einige Funktionen ergänzt.

Das Programm liest aus *text.txt* (siehe Datei 12) die aktuellen Messwerte aus. Da der Platz jedoch stark beschränkt ist, werden diese auf 11 Seiten aufgeteilt, zwischen denen das Display alle 3 Sekunden wechselt.



Abbildung 3.1.: eingebautes Display

## 3.3. Webinterface

Einer der wichtigsten Teile des Projektes ist die grafische Auswertung. Diese kann live auf der Webseite des Raspberry Pi und zeitverzögert unter [winkler.kremszeile.at](http://winkler.kremszeile.at) angesehen werden. Die Auswertung besteht aus zwei voneinander unabhängigen Teilen. Zum einen gibt es die Anzeige der Live-Daten, zum anderen die Darstellung der kompletten Aufzeichnung als interaktives Diagramm. Zusätzlich werden auf zwei zusätzlichen Seiten

---

<sup>24</sup>Schnatterente.net, [displaytest.py](http://schnatterente.net/displaytest.py).

### 3. Software

Informationen über das Projekt und Links zu anderen Projektseiten von mir (siehe Anhang A) angezeigt.

#### 3.3.1. Livedaten

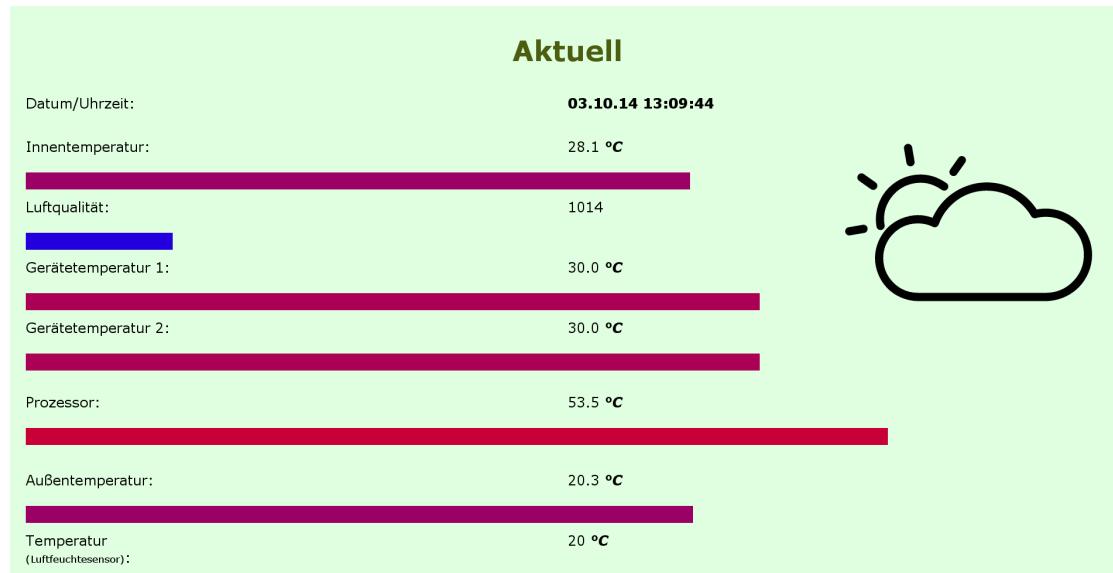


Abbildung 3.2.: Livedaten (Screenshot von [winkler.kremszeile.at/aktuell.html](http://winkler.kremszeile.at/aktuell.html))

Auf Webinterface können die aktuellen Messwerte angezeigt und grafisch veranschaulicht werden.<sup>25</sup> Hierzu wird alle 5 Sekunden mithilfe von JavaScript die Datei *text\_ws.txt* nachgeladen und ausgewertet. Zusätzlich zur Anzeige der Zahlenwerte werden die Messungen mithilfe von Balken und Farbverläufen angezeigt. Für ältere Webbrowser gibt es auch eine einfache tabellarische Ansicht<sup>26</sup>

#### Wetter-Rater

Um die Daten auch anders zu nutzen habe ich einen *Wetter-Rater* programmiert. Dieser versucht auf Basis von einfachen Berechnungen und Schätzungen das

<sup>25</sup>[winkler.kremszeile.at/aktuell.html](http://winkler.kremszeile.at/aktuell.html)

<sup>26</sup>[winkler.kremszeile.at/aktuell\\_einfach.html](http://winkler.kremszeile.at/aktuell_einfach.html)

### 3. Software

aktuelle Wetter zu „erraten“. So wird zum Beispiel die aktuelle Außentemperatur mit der nach Jahreszeit und Tageszeit zu erwartenden Temperatur verglichen, um Rückschlüsse auf den Bewölkungsgrad zu ziehen oder aufgrund der Luftfeuchtigkeit ermittelt, ob es Niederschlag gibt.

Hierzu wird die Temperaturschwankung über einen Tag als Cosinusfunktion mit einer Schwankung von 5 °C angenommen.

$$\text{Temperaturerwartung} = -5 \cdot \cos\left(\frac{\text{Stunde} \cdot 2 \cdot \pi}{24}\right) + \text{Mittlere Temp. des Tages};$$

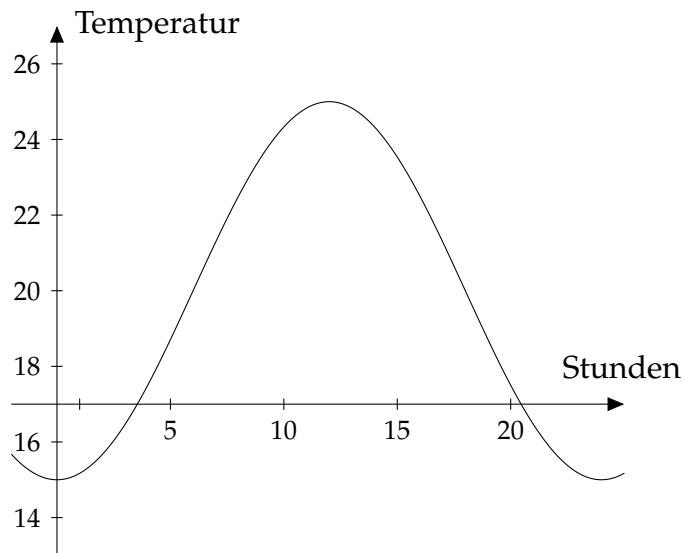


Abbildung 3.3.: Graph der Funktion bei mittlere Temp. = 20 °C

Der Wetter-Rater gibt drei Werte aus:

- Tag oder Nacht
- Niederschlag (keiner/Gewitter/Schnee/Regen/Schneeregen)
- Bewölkungsgrad (sonnig/leicht bewölkt/stark bewölkt)

Aufgrund von diesen Werten wird dann eine Grafik (siehe Abbildung 3.4) ausgewählt, welche dann angezeigt wird.

### 3. Software



Abbildung 3.4.: Wettericons

#### 3.3.2. Diagramme

Damit auch die vergangenen Messergebnisse angesehen werden können, werden diese im Webinterface auf einer eigenen Seite als Diagramm dargestellt. Diese werden mithilfe von *dygraphs*<sup>27</sup>, einer *JavaScript*-Bibliothek für interaktive Diagramme, erstellt.<sup>28</sup>

Hierzu lädt der Webbrower die komplette Aufzeichnung in Form der Datei *dygraphs.csv* (siehe Datei 10) nach. Daraus wird ein Diagramm über den gesamten Zeitraum der Messung erstellt. Im Gegensatz zu anderen Darstellungen kann hier jedoch einfach hineingezoomt werden. So können auch einzelne Wochen oder Tage betrachtet werden. Da das Diagramm mit allen 10 Messkurven auf einmal überladen wäre, können einzelne Kurven aktiviert und deaktiviert werden. Zusätzlich kann ein Faktor eingegeben werden, um den die Kurven automatisch geglättet werden sollen. Mit einem Tastendruck kann man auch auf die letzten 24 Stunden oder 7 Tage zoomen oder in einem Kalender den Zeitraum auswählen. Es gibt auch ein Auswahlmenü, um den gewünschten Datensatz auszuwählen. Das Diagramm ist darauf angepasst, auch mit Tablets bzw. Touchscreens bedient zu werden. Auf Youtube gibt es ein Video, wo man die Verwendung in Aktion sehen kann: [www.youtube.com/watch?v=1bv6CEXuN5c](http://www.youtube.com/watch?v=1bv6CEXuN5c)

---

<sup>27</sup>dygraphs, *Homepage*.

<sup>28</sup>[winkler.kremszeile.at/dygraphs\\_aussen.html#dygraph8A.csv](http://winkler.kremszeile.at/dygraphs_aussen.html#dygraph8A.csv)

### 3. Software



Abbildung 3.5.: Webinterface auf einem Tablet

## 3.4. Endauswertung

Unabhängig vom Webinterface habe ich ein zweites Programm in [Python](#) geschrieben. Dieses kann die fertige *dygraphs.csv* (siehe Datei [10](#)) einlesen und verschiedene mathematische Auswertungen über einen beliebigen Zeitraum erstellen.

Zunächst wird die Datei in zwei Dateien aufgespalten. Die eine enthält nur die erste Spalte mit den Zeitstempeln jeder Messung, die andere die Messergebnisse. Da schon wenige Messfehler (zum Beispiel einmalig 6 °C bei 20 °C Außentemperatur) den Mittelwert (und die [Standardabweichung](#)) stark

### 3. Software

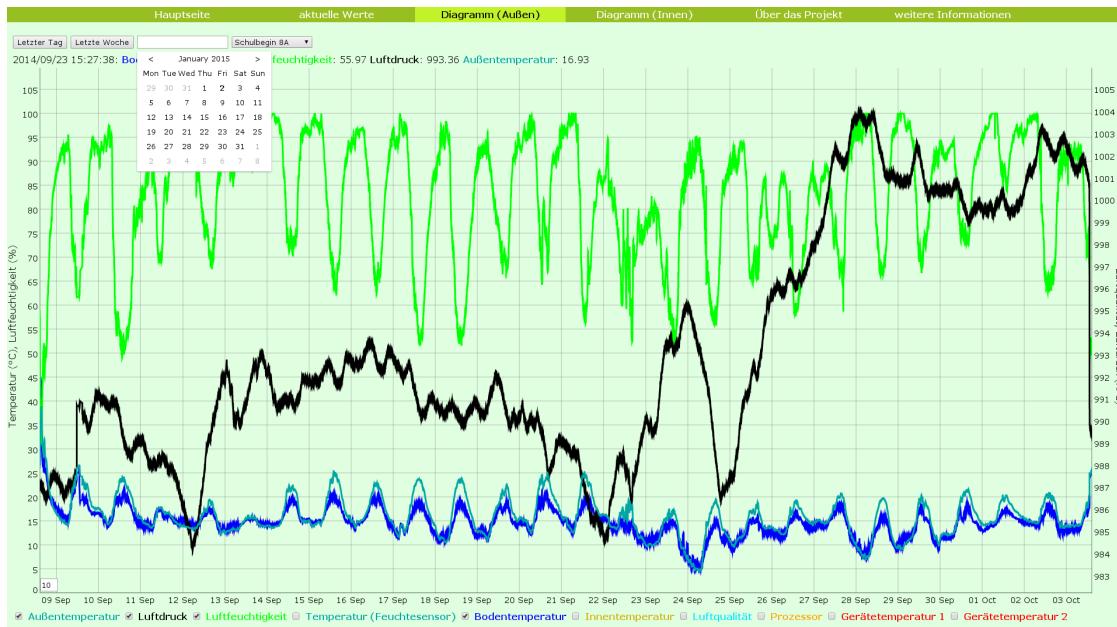


Abbildung 3.6.: Webinterface: [winkler.kremszeile.at/dygraphs\\_aussen.html#dygraph8A.csv](http://winkler.kremszeile.at/dygraphs_aussen.html#dygraph8A.csv)

verändern, wird zunächst nach Ausreißern gesucht. Hierzu wird für jeden Messwert überprüft, ob er um mehr als  $10^{\circ}\text{C}$  von dem vorherigen *und* dem folgenden abweicht. Wenn dem so ist, wird eine Meldung ausgegeben:

```
5 in Spalte 5 Zeile 43636 ist ein Ausreisser (25.5)
6 in Spalte 6 Zeile 6828 ist ein Ausreisser (5.0)
7 in Spalte 6 Zeile 30728 ist ein Ausreisser (87.2)
8 in Spalte 10 Zeile 885 ist ein Ausreisser (0.0)
```

Datei 14: ausgabe.txt (Zeile 5 bis 8)

Als nächstes kann man den Zeitraum angeben, für den die Auswertung erstellt werden soll. Zunächst wird überprüft, ob das eingegebene Datum gültig ist. Das Programm testet anschließend, welche Zeilen der Eingabedatei in diesem Zeitraum liegen.

```
22 Bitte Datum im Format 'DD.MM.YY HH:MM:SS' eingeben
23 Es sollte zwischen 22.02.14 10:24:30 und 13.03.14 14:54:50
     liegen
24 von: 22.02.14 10:30:00
25 bis: 30.02.14 00:00:00
```

### 3. Software

```
26 Bitte Datum im Format 'DD.MM.YY HH:MM:SS' eingeben  
27 bis: 13.03.14 14:00:00  
28 Der Messwert geht von Zeile 14 bis Zeile 50353 und über  
folgenden Zeitraum: 19 days, 3:30:00
```

Datei 15: ausgabe.txt (Zeile 22 bis 28)

Nun werden für jede Spalte bzw. für jeden Sensor der Mittelwert, das Minimum, das Maximum und die **Standardabweichung** berechnet und ausgegeben:

```
29 -----Mittelwerte-----  
30 Innentemperatur: 24.40  
31 Gerätetemperatur 1: 26.34  
32 Außentemperatur: 9.38  
33 Gerätetemperatur 2: 26.34  
34 Temperatur (Luft): 11.18  
35 Luftfeuchtigkeit: 50.90  
36 Luftdruck: 989.33  
37 Temperatur (Druck): 11.31  
38 Prozessor: 51.92  
39 Qualität: 2154.99  
40 -----Minimum-Maximum-----  
41 Innentemperatur: 17.187 26.437  
42 Gerätetemperatur 1: 19.687 30.5  
43 Außentemperatur: 0.687 32.687  
44 Gerätetemperatur 2: 19.687 30.5  
45 Temperatur (Luft): 1.1 25.5  
46 Luftfeuchtigkeit: 0.0 89.5  
47 Luftdruck: 966.99 1005.38  
48 Temperatur (Druck): 1.2 24.4  
49 Prozessor: 32.0 57.8  
50 Qualität: 0.0 6001.0  
51 -----Standardabweichung-----  
52 Innentemperatur: 0.90  
53 Gerätetemperatur 1: 0.94  
54 Außentemperatur: 6.33  
55 Gerätetemperatur 2: 0.94  
56 Temperatur (Luft): 6.55  
57 Luftfeuchtigkeit: 19.34  
58 Luftdruck: 11.31
```

### 3. Software

59 | Temperatur (Druck) : -6.48

Datei 16: ausgabe.txt (Zeile 29 bis 59)

## 3.5. diverses

Abgesehen von den großen Teilen des Projektes gibt es auch kleinere Aspekte, von denen ich nun einige vorstellen möchte:

### 3.5.1. Weather Underground

*Weather Underground* ist ein Online-Wetterdienst mit Firmensitz in San Francisco.<sup>29</sup> Dieser bietet auch die Möglichkeit eine eigene Wetterstation zu betreiben und die Daten auf ihrer Webseite anzuzeigen.<sup>30</sup> Leider werden die Wetterdaten nur in imperialen Einheiten (also °F und mmHg<sup>31</sup>) akzeptiert. Daher werden die Messwerte vorher von einem [Python](#)-Programm umgerechnet und anschließend hochgeladen.

### 3.5.2. Autostart

Damit die Umweltdatenmessung einfacher zu handhaben ist, gibt es ein einfaches Start-/Stop-Skript. So werden die Aufzeichnung und das Display automatisch beim Hochfahren gestartet und vor dem Herunterfahren ordnungsgemäß beendet.

### 3.5.3. mitternacht.sh

Da der Raspberry Pi in der Schule hängt und nicht von außerhalb erreicht werden kann, ist es schwierig Softwareänderungen anzuwenden. Daher habe ich das kleine Skript *mitternacht.sh* geschrieben. Dieses startet täglich um 0:00

---

<sup>29</sup>Wikipedia, [Weather Underground \(weather service\)](#) — Wikipedia, The Free Encyclopedia.

<sup>30</sup>[www.wunderground.com/personal-weather-station/dashboard?ID=INIEDERS353](http://www.wunderground.com/personal-weather-station/dashboard?ID=INIEDERS353)

<sup>31</sup>Millimeter-Quecksilbersäule oder Torr

### 3. Software

Uhr und lädt die neuerste Version von [Github](#) und installiert diese. Zusätzlich werden auch das Betriebssystem und alle installierten Programme aktualisiert. Anschließend werde ich per Push-Benachrichtigung über etwaige Probleme benachrichtigt und der Raspberry Pi startet neu.

#### 3.5.4. sonstiges

- ein Logo (siehe Abbildung 3.7)
- Erstellung von statischen Diagrammen mit [Gnuplot](#)
  - Jedoch bei zu vielen Daten unübersichtlich und langsam
- E-Mail und Push-Benachrichtigungen
- regelmäßiger Upload der Daten auf [winkler.kremszeile.at](http://winkler.kremszeile.at)

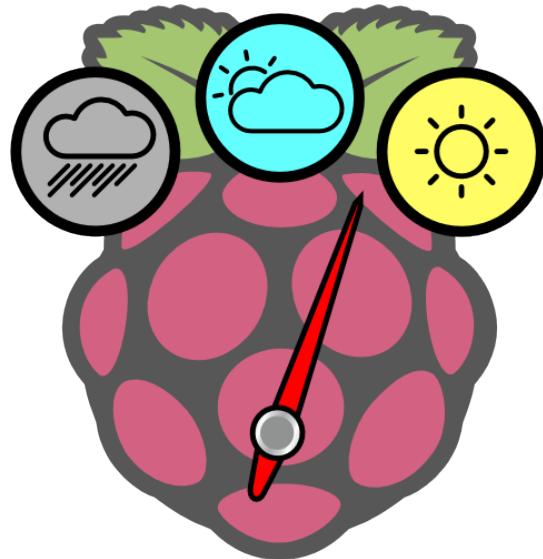


Abbildung 3.7.: Logo

## 4. Auswertung

Nun möchte ich die Auswertung der Daten anhand von einer mehrwöchigen Aufzeichnung im Klassenzimmer demonstrieren.

### 4.1. Aufzeichnung

Zu Beginn des Schuljahrs 2014/15 habe ich meine Messstation am 8.9.2014 in der Klasse aufgebaut. Die Messung ging mit nur einigen Minuten Unterbrechung bis zum 3.10.2014. Hierbei wurden ungefähr 2 Mal in der Minute 10 Sensoren ausgelesen. Die dabei entstandene [CSV-Datei](#), hat 75875 Zeilen und ist knapp über 6 MB groß.<sup>32</sup>

### 4.2. Graphische Darstellung

Die Messdaten lassen sich sehr einfach auswerten, wenn man sie als Diagramme darstellt (siehe [3.3.2](#)).

So merkt man bei den meisten Sensoren, dass die Werte alle 24 Stunden periodisch schwanken. Am besten ist das bei der Luftfeuchtigkeit zu erkennen, welche zwischen ca. 60 % (Mittag bis Abend) und 100 % rel LF (Mitternacht bis Vormittag) schwankt. (siehe Abbildungen [4.1](#) und [4.2](#)). Da das Klassenzimmer außerhalb des Unterrichts nicht geheizt wird und die Wände des Containers kaum isolieren, schwankt auch die Innentemperatur an Schultagen sehr stark. Verstärkt wird dies dadurch, dass der Temperatursensor auf der Seite des Raumes befestigt war, an dem auch die Fenster und Heizkörper sind. Es ist auch erkennbar, dass es an Wochenenden deutlich kälter ist, da nicht geheizt

---

<sup>32</sup>Sie kann unter [winkler.kremszeile.at/dygraph\\_8A.csv](http://winkler.kremszeile.at/dygraph_8A.csv) heruntergeladen werden.

#### 4. Auswertung

wird. Erstaunlich ist jedoch, dass die Temperaturschwankungen sogar an der CPU-Temperatur deutlich erkennbar sind. (siehe Abbildung 4.3)

Am ungenauesten sind die Ergebnisse des Luftqualitätssensors. Diese enthalten viele Ausreißer und sind im Allgemeinen viel zu hoch. Die Originalsoftware gibt in den Standardeinstellungen bei einem Wert von 1500 die Warnung „Bad air quality“ an. Im Klassenzimmer wurden jedoch häufig Werte über 2000 gemessen. (siehe Abbildung 4.4) Erkennbar ist jedoch, dass die Luftqualität erwartungsgemäß im Laufe des Vormittags immer schlechter wird und sich nachmittags langsam wieder erholt.

#### 4. Auswertung

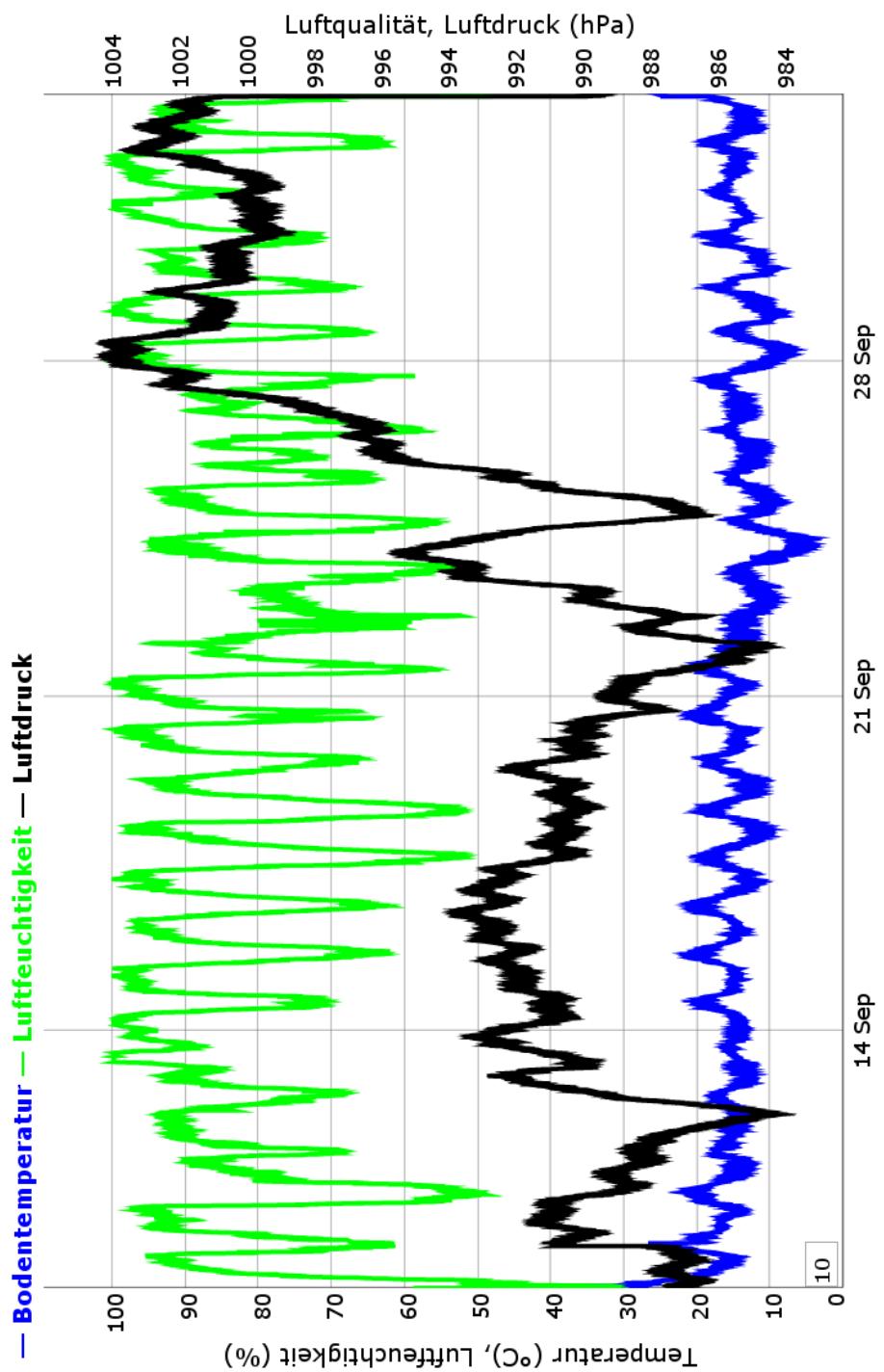


Abbildung 4.1.: Außensensoren

#### 4. Auswertung

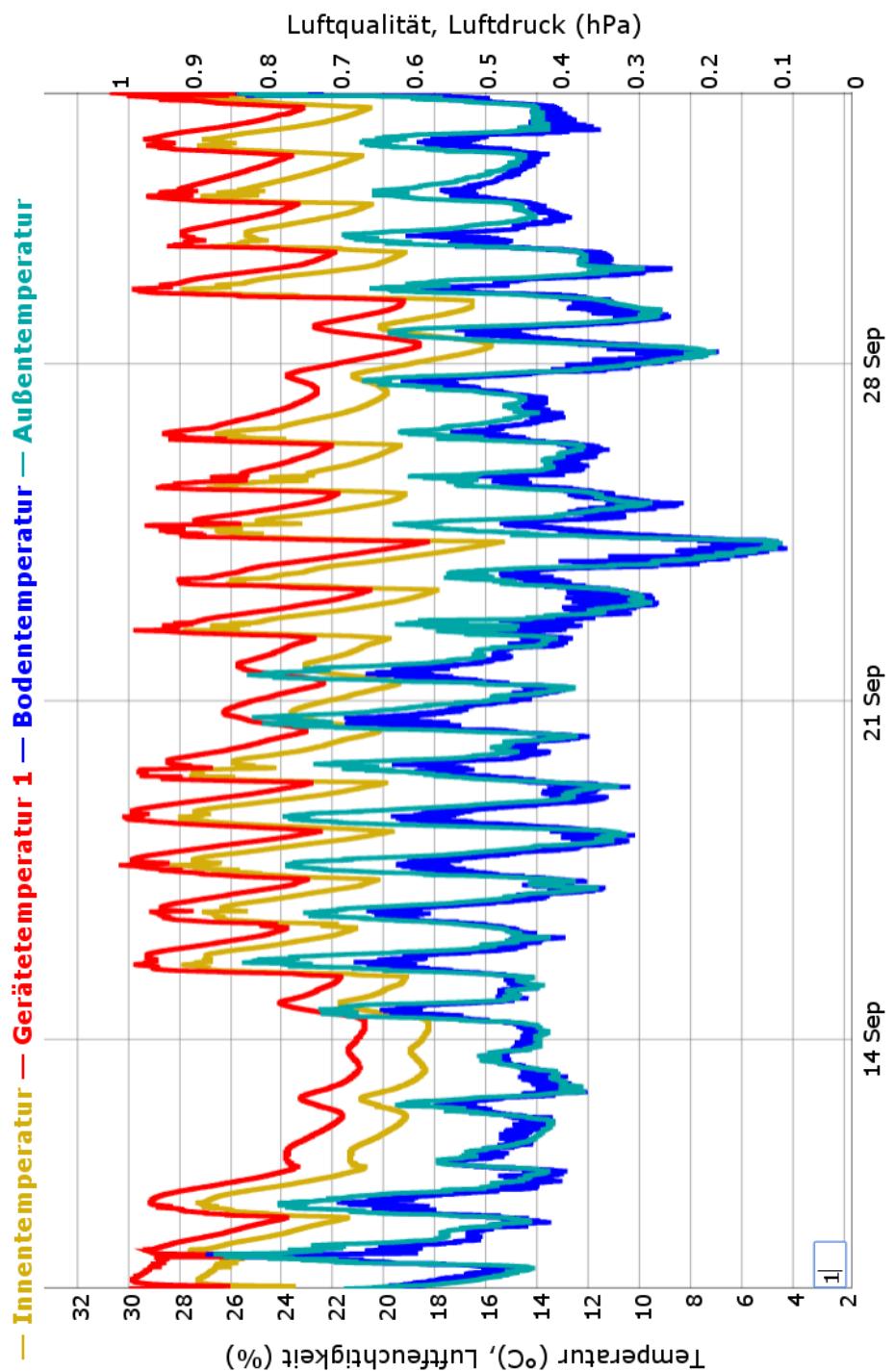


Abbildung 4.2.: Temperatursensoren

#### 4. Auswertung

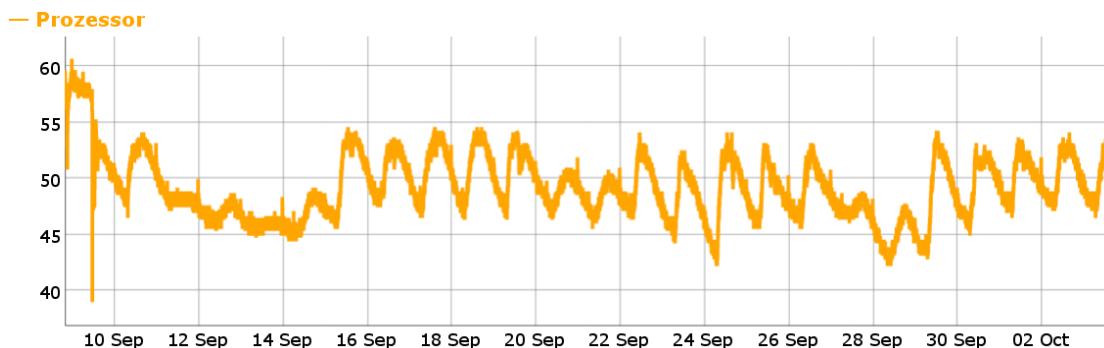


Abbildung 4.3.: CPU-Temperatur

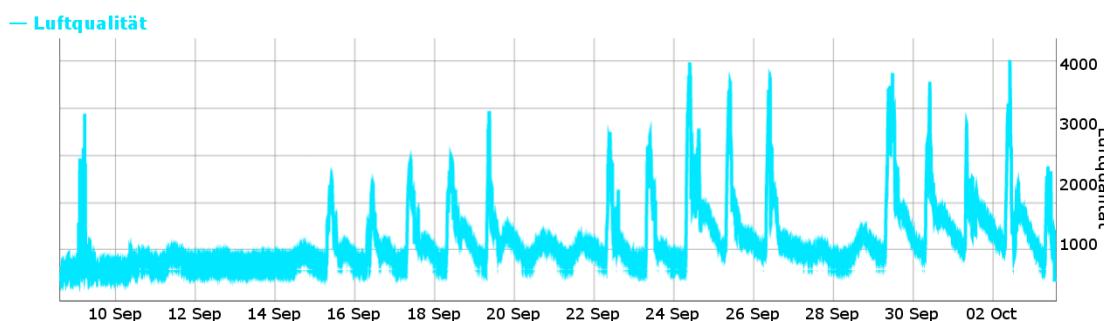


Abbildung 4.4.: Luftqualität

### 4.3. Endauswertung

Sensor	Einheit	Mittelwert	Minimum	Maximum	Standardabweichung
Innentemperatur	°C	22.44	15.375	36.437	2.97
Gerätetemperatur 1	°C	25.01	18.312	38.375	2.85
Gerätetemperatur 2	°C	25.02	18.25	38.437	2.85
Bodentemperatur	°C	14.73	4.312	44.687	3.05
Außentemperatur	°C	15.94	4.5	39.0	3.96
Außentemperatur 2	°C	15.87	3.5	38.7	3.95
Luftfeuchtigkeit	% rel. LF	82.95	14.7	99.9	12.68
Luftdruck	hPa	993.26	984.04	1004.12	5.27
Prozessor	°C	49.34	39.0	62.7	3.07
Qualität	rel. Wert	1026.37	450.0	5870.0	529.91

## 5. Fazit

Ich habe im letzten Jahr zunächst durch das Projekt und anschließend durch das Schreiben darüber einiges gelernt. Abgesehen von den technischen Details, lernte ich mich längere Zeit mit einem Thema zu beschäftigen und das Thema von verschiedenen Perspektiven (Messung, Auswertung, Darstellung) zu betrachten und bearbeiten.

Dec 22, 2013 – Feb 2, 2015

Contributions to master, excluding merge commits

Contributions: **Commits** ▾

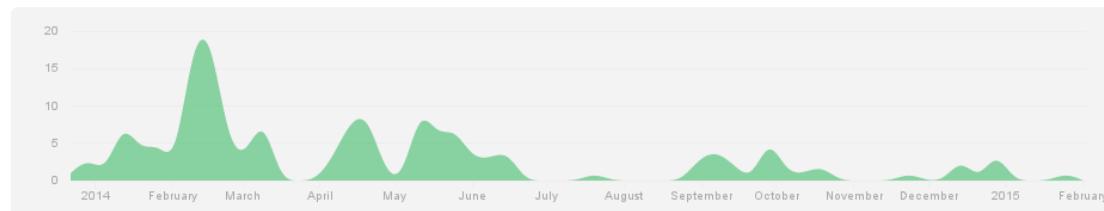


Abbildung 5.1.: Verlauf auf Github

Eine andere Erkenntnis aus dem Projekt ist, dass aus einer kleinen Idee ein ziemlich umfangreiches Projekt werden kann, welches sogar einen Preis gewinnt. (siehe Anhang B) Angefangen hat alles am Beginn des Schuljahres 2013/14, als wir Ideen für ein *Raspberry Pi*-Projekt recherchierten sollten. Ich hatte in den Ferien schon mit einem experimentiert und kam auf die Idee eine *Wetterstation* zu bauen. Eine Woche später war das erste Programm fertig, welches Zufallszahlen, Prozessortemperatur und -auslastung mithilfe von Gnuplot als Diagramm anzeigt.<sup>33</sup>

---

<sup>33</sup>hier kann es noch gesehen werden: [gist.github.com/Findus23/d1187031f875b76a69e8](https://gist.github.com/Findus23/d1187031f875b76a69e8) (`csv2gnuplot.sh` ist nicht von mir)

## 5. Fazit

Durch das Schreiben dieser VWA und den Präsentationen (Anhang B) habe ich gelernt technische Details so weit wie möglich allgemeinverständlich zu erklären und versuche bei den Präsentationen diese kurz zu fassen.

### 5.1. Ausblick

Wie es mit dem Projekt weitergehen wird, weiß ich noch nicht. Es gebe jedoch noch sehr viele Möglichkeiten zur Erweiterung. Vor allem können die Daten genutzt werden um andere Geräte zu steuern. So könnte der *Raspberry Pi* zum Beispiel eine Heizung oder Beleuchtung ein- und ausschalten oder bei nicht optimaler Luftqualität im Klassenzimmer eine Warnung abgeben.

Seit ich begonnen habe, sind einige neue Modelle des *Raspberry Pi* herausgekommen, mit denen man mehr Sensoren hinzufügen könnte (zum Beispiel Helligkeit oder Lautstärke) oder die leistungsstärker sind und ohne zusätzliche Geräte eine grafische Darstellung auf einem Display anzeigen können.

# Anhang A.

## Weitere Informationen

In dieser VWA können nur die wichtigsten Teile des Projektes erwähnt werden. Weiterführende Informationen gibt es unter folgenden Quellen:

- [Github: `github.com/Findus23/Umweltdatenmessung`](https://github.com/Findus23/Umweltdatenmessung)
  - Der komplette Programmcode und alle anderen Dateien, die zum Projekt gehören, sind hier gesammelt.
  - Auch alle Veränderungen seit Dezember 2013 können hier angesehen werden: <https://github.com/Findus23/Umweltdatenmessung/commits/master>
  - Große Veränderungen werden zusätzlich separat gelistet: <https://github.com/Findus23/Umweltdatenmessung/releases>
- [Flickr: `www.flickr.com/photos/findus23/sets/72157637721138445`](https://www.flickr.com/photos/findus23/sets/72157637721138445)
  - Hier sind über 100 Bilder vom Projekt zu sehen.
- [Youtube: `https://www.youtube.com/playlist?list=PLjtFdocVknd4aw90\_zVr0U4BF1RH9PatA`](https://www.youtube.com/playlist?list=PLjtFdocVknd4aw90_zVr0U4BF1RH9PatA)
  - Einige Videos (z.B. vom Display) sind auf Youtube zu finden.

## Anhang B.

### Präsentationen

Während ich am Projekt arbeitete, hatte ich mehrmals die Möglichkeit es anderen vorzustellen. So konnte ich zum Beispiel am 23. April 2014 bei den EDU|days<sup>34</sup> den *Raspberry Pi – Anfänger/innen Workshop* von meinem Klassenvorstand MMag. Rene Schwarzinger begleiten und dort den aktuellen Zwischenstand präsentieren.

Nach dem Workshop sprach mich Dr. Johann Stockinger an und fragte mich, ob ich beim computer creative wettbewerb<sup>35</sup> teilnehmen möchte. Noch in derselben Woche habe ich einen kurzen Text über mein Projekt<sup>36</sup> geschrieben und eingereicht.

Ein Monat später erfuhr ich, dass ich im Finale bin<sup>37</sup> und daher am 17. Juni 2014 mein Projekt vor einer Jury präsentieren darf.

Nach einem langen Tag mit vielen Präsentationen erfuhr ich am Nachmittag: Ich habe den ersten Preis in der Sekundarstufe II erreicht.<sup>38</sup> Anschließend schrieb ich im Sommer einen Artikel für das OCG Journal.<sup>39</sup>

Am 6. Oktober 2014 durfte ich mein Projekt der Arbeitsgruppe *Bildung, Wissenschaft und Forschung* am 3. IKT-Konvent präsentieren.<sup>40</sup>

---

<sup>34</sup>[www.edudays.at/index.php/programm2014](http://www.edudays.at/index.php/programm2014)

<sup>35</sup>[www.ocg.at/de/computer-creative-wettbewerb](http://www.ocg.at/de/computer-creative-wettbewerb)

<sup>36</sup>[https://github.com/Findus23/Umweltdatenmessung/blob/master/Dokumentationen/OCG\\_Wettbewerb.pdf?raw=true](https://github.com/Findus23/Umweltdatenmessung/blob/master/Dokumentationen/OCG_Wettbewerb.pdf?raw=true)

<sup>37</sup>[blog.ocg.at/2014/05/ccw14-finale/](http://blog.ocg.at/2014/05/ccw14-finale/)

<sup>38</sup>[blog.ocg.at/2014/06/ccw14-final/](http://blog.ocg.at/2014/06/ccw14-final/)

<sup>39</sup>OCG Journal 3/2014: Seite 33 ([www.ocg.at/sites/ocg.at/files/medien/pdfs/OCG-Journal1403.pdf](http://www.ocg.at/sites/ocg.at/files/medien/pdfs/OCG-Journal1403.pdf))

<sup>40</sup>[www.internetoffensive.at/3-ikt-konvent](http://www.internetoffensive.at/3-ikt-konvent)

WERKUNDE



Lukas Winkler

hat beim

## computer creative wettbewerb '14

der Österreichischen Computer Gesellschaft  
mit dem Projekt

Wetterdatenerfassung mit Raspberry Pi

in der Kategorie Sekundarstufe II

1. Platz

erreicht.



für die Jury  
Dr. Christian Wirth

Wien, Juni 2014

## Anhang B. Präsentationen



Abbildung B.1.: Präsentation im Finale vom computer creative wettbewerb

# Literatur

- Adafruit User KEVIN TOWNSEND. *Using the BMP085/180 with Raspberry Pi or Beaglebone Black*. 2013. URL: <https://learn.adafruit.com/using-the-bmp085-with-raspberry-pi?view=all> (besucht am 25.10.2014) (siehe S. 7, 15).
- Adafruit User LADY ADA. *DHT Humidity Sensing on Raspberry Pi or Beaglebone Black with GDocs Logging*. 2013. URL: <https://learn.adafruit.com/dht-humidity-sensing-on-raspberry-pi-with-gdocs-logging?view=all> (besucht am 25.10.2014) (siehe S. 6, 15).
- ANAG VISION. *AV1624 Datasheet*. 22. Mai 2006. URL: [http://conrad.ru/doci/tekstovyy\\_display\\_stn\\_anag\\_vision\\_av1624gfbw\\_sj\\_\\_seryy\\_181664\\_en.pdf](http://conrad.ru/doci/tekstovyy_display_stn_anag_vision_av1624gfbw_sj__seryy_181664_en.pdf) (besucht am 23.11.2014) (siehe S. 8).
- Aosong Electronics Co.,Ltd. *Digital-output relative humidity & temperature sensor/module DHT22*. 2011. URL: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf> (besucht am 08.11.2014) (siehe S. 6).
- Bosch Sensor tec. *BMP085 Digital pressure Sensor - Data Sheet*. 15. Okt. 2009. URL: <https://www.sparkfun.com/datasheets/Components/General/BST-BMP085-DS000-05.pdf> (besucht am 25.10.2014) (siehe S. 6).
- dygraphs. *Homepage*. 2014. URL: <http://dygraphs.com/> (besucht am 19.12.2014) (siehe S. 21).
- elinux. *RPi Hardware - Power*. 2014. URL: [http://elinux.org/index.php?title=RPi\\_Hardware&oldid=341192#Power](http://elinux.org/index.php?title=RPi_Hardware&oldid=341192#Power) (besucht am 04.07.2014) (siehe S. 4).
- FHEMWiki. *Kategorie:1-Wire - FHEMWiki*. 2014. URL: <http://www.fhemwiki.de/w/index.php?title=Kategorie:1-Wire&oldid=5092#1-Wire> (besucht am 18.10.2014) (siehe S. 5).
- Innenraumlufthygiene-Kommission des Umweltbundesamtes. *LEITFÄDEN FÜR DIE INNENRAUMHYGIENE IN SCHULGEBÄUDEN*. 2008. URL: <http://www.umweltbundesamt.de/sites/default/files/medien/publikation/long/3689.pdf> (besucht am 11.11.2014) (siehe S. 7).
- Maxim Integrated Products, Inc. *DS18B20 - Data Sheet*. 2008. URL: <http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf> (besucht am 17.10.2014) (siehe S. 5).

## Literatur

- Pollin.de. *LCD-Modul TC1602E-01*. 2014. URL: [http://www.pollin.de/shop/dt/OTc10Tc40Tk-/Bauelemente\\_Bauteile/Aktive\\_Bauelemente/Displays/LCD\\_Modul\\_TC1602E\\_01.html](http://www.pollin.de/shop/dt/OTc10Tc40Tk-/Bauelemente_Bauteile/Aktive_Bauelemente/Displays/LCD_Modul_TC1602E_01.html) (besucht am 23. 11. 2014) (siehe S. 9).
- Python Software Foundation. *What is Python? Executive Summary*. URL: <https://www.python.org/doc/essays/blurb/> (besucht am 02. 01. 2015) (siehe S. 47).
- Raspberry Pi Foundation. *The Making of Pi*. Raspberry Pi Foundation. 2012. URL: <http://www.raspberrypi.org/about/> (besucht am 04. 07. 2014) (siehe S. 4).
- @Raspberry\_Pi. "@ruskin147 As of today, it looks like 3.8 million - that's an \*awful lot of computers\*." 12. Okt. 2014. URL: [https://twitter.com/Raspberry\\_Pi/status/521065388948586497](https://twitter.com/Raspberry_Pi/status/521065388948586497) (besucht am 22. 11. 2014) (siehe S. 4).
- Rossum, Guido van. *Python Reference Manual*. 31. Dez. 1997. URL: <http://svn.python.org/projects/python/tags/release15/Doc/ref/ref.ps> (besucht am 02. 01. 2015) (siehe S. 47).
- Schnatterente.net. *displaytest.py*. URL: <http://www.schnatterente.net/code/raspberrypi/displaytest.py> (besucht am 13. 12. 2014) (siehe S. 18).
- *Raspberry Pi: 32 Zeichen Hitachi HD44780 Display*. 10. Okt. 2014. URL: <http://www.schnatterente.net/technik/raspberry-pi-32-zeichen-hitachi-hd44780-display#> (besucht am 23. 11. 2014) (siehe S. 8).
- Tfitzp. *Veroboard sample*. 22. Aug. 2013. URL: [http://commons.wikimedia.org/wiki/File:VEROBOARD\\_sample.jpg](http://commons.wikimedia.org/wiki/File:VEROBOARD_sample.jpg) (besucht am 05. 01. 2015) (siehe S. 47).
- Ulfbastel. *gold plated printed circuit board*. 22. Aug. 2007. URL: <http://commons.wikimedia.org/wiki/File:Aupcb.jpg> (besucht am 05. 01. 2015) (siehe S. 46).
- usb-sensors-linux. *Install AirSensor on Linux*. 29. Apr. 2013. URL: [https://code.google.com/p/usb-sensors-linux/wiki/Install\\_AirSensor\\_Linux](https://code.google.com/p/usb-sensors-linux/wiki/Install_AirSensor_Linux) (besucht am 08. 11. 2014) (siehe S. 8, 16).
- Velux. *VELUX Raumluftfühler*. 2014. URL: [http://www.velux.de/privatkunden/produkte/integra\\_system/produkte/produktempfehlung/raumluftfuehler](http://www.velux.de/privatkunden/produkte/integra_system/produkte/produktempfehlung/raumluftfuehler) (besucht am 08. 11. 2014) (siehe S. 8).
- Wikipedia. *Bus — Wikipedia, Die freie Enzyklopädie*. 2014. URL: [http://de.wikipedia.org/w/index.php?title=Bus\\_\(Datenverarbeitung\)&oldid=134938136](http://de.wikipedia.org/w/index.php?title=Bus_(Datenverarbeitung)&oldid=134938136) (besucht am 04. 07. 2014) (siehe S. 44).
- *GitHub — Wikipedia, Die freie Enzyklopädie*. 12. Dez. 2014. URL: <http://de.wikipedia.org/w/index.php?title=GitHub&oldid=136725990> (besucht am 14. 12. 2014) (siehe S. 45).
- *Gnuplot — Wikipedia, Die freie Enzyklopädie*. 12. Jan. 2015. URL: <http://de.wikipedia.org/w/index.php?title=Gnuplot&oldid=137696684> (besucht am 02. 02. 2015) (siehe S. 45).

## Literatur

- Wikipedia. *Ohm* — Wikipedia, Die freie Enzyklopädie. 8. Dez. 2014. URL: <http://de.wikipedia.org/w/index.php?title=Ohm&oldid=136576262> (besucht am 22.12.2014) (siehe S. 46).
- *Raspberry Pi*— Wikipedia, Die freie Enzyklopädie. 2014. URL: [http://de.wikipedia.org/w/index.php?title=Raspberry\\_Pi&oldid=134104012#Idee](http://de.wikipedia.org/w/index.php?title=Raspberry_Pi&oldid=134104012#Idee) (besucht am 04.07.2014) (siehe S. 4).
  - *Weather Underground (weather service)* — Wikipedia, The Free Encyclopedia. 12. Dez. 2014. URL: [http://en.wikipedia.org/w/index.php?title=Weather\\_Underground\\_\(weather\\_service\)&oldid=637705817](http://en.wikipedia.org/w/index.php?title=Weather_Underground_(weather_service)&oldid=637705817) (besucht am 03.01.2015) (siehe S. 25).
- WISSEN Wiki. *Flüchtige organische Verbindung*. 2014. URL: [http://www.wissenwiki.de/index.php?title=Fl%C3%BCchtige\\_organische\\_Verbindung&oldid=41478](http://www.wissenwiki.de/index.php?title=Fl%C3%BCchtige_organische_Verbindung&oldid=41478) (besucht am 11.11.2014) (siehe S. 7).

# Abbildungsverzeichnis

1.1.	Messstation	2
2.1.	Raspberry Pi - Modell B	3
2.2.	Pinbelegung des DS18B20	5
2.3.	Die erste erfolgreiche Messung	6
2.4.	Anschlusskizze von <i>DS18B20</i> (Mitte; 2.2.1), <i>DHT22</i> (Links; 2.2.2) und <i>BMP085</i> (Rechts; 2.2.3)	7
2.5.	Velux Raumluftfühler	8
2.6.	Das erste Display	9
2.7.	<i>DS18B20</i> (2.2.1)	10
2.8.	Steckbrett mit Display (2.3)	10
2.9.	erste mehrwöchige Messung	10
2.10.	neue Sensoren auf dem Steckbrett	10
2.11.	Außensensoren	11
2.12.	Innensensoren	11
2.13.	Steckverbindung am Display	11
2.14.	Hauptplatine	11
2.15.	Komplette Hardware	11
2.16.	Die Hardware in der Holzbox	12
3.1.	eingebautes Display	18
3.2.	Livedaten (Screenshot von <a href="http://winkler.kremszeile.at/aktuell.html">winkler.kremszeile.at/aktuell.html</a> )	19
3.3.	Graph der Funktion bei mittlere Temp. = 20 °C	20
3.4.	Wettericons	21
3.5.	Webinterface auf einem Tablet	22
3.6.	Webinterface	23
3.7.	Logo	26
4.1.	Außensensoren	29
4.2.	Temperatursensoren	30
4.3.	CPU-Temperatur	31

## Abbildungsverzeichnis

4.4. Luftqualität . . . . .	31
5.1. Verlauf auf Github . . . . .	32
B.1. Präsentation im Finale vom computer creative wettbewerb . . . . .	37

# Dateiverzeichnis

1.	main.sh (Zeile 8 bis 13) . . . . .	13
2.	main.sh (Zeile 27 bis 28) . . . . .	14
3.	main.sh (Zeile 29 bis 31) . . . . .	14
4.	main.sh (Zeile 32) . . . . .	15
5.	main.sh (Zeile 33 bis 40) . . . . .	15
6.	main.sh (Zeile 4) . . . . .	16
7.	main.sh (Zeile 66 bis 69) . . . . .	16
8.	main.sh (Zeile 84) . . . . .	16
9.	main.sh (Zeile 89 bis 90) . . . . .	16
10.	dygraphs.csv . . . . .	17
11.	main.sh (Zeile 92) . . . . .	17
12.	text.txt . . . . .	17
13.	text_ws.txt . . . . .	17
14.	ausgabe.txt (Zeile 5 bis 8) . . . . .	23
15.	ausgabe.txt (Zeile 22 bis 28) . . . . .	23
16.	ausgabe.txt (Zeile 29 bis 59) . . . . .	24

# Glossar

[A](#) | [B](#) | [C](#) | [D](#) | [E](#) | [F](#) | [G](#) | [H](#) | [I](#) | [J](#) | [K](#) | [L](#) | [O](#) | [P](#) | [S](#) | [V](#)

**A**

**Ampere**

die SI-Basiseinheit der elektrischen Stromstärke

**B**

**Bash**

*Bourne-again shell*

ist die heute unter Linux am häufigsten verwendete [Shell](#)

**C**

**C**

eine sehr weit verbreitete Programmiersprache

Hier wird sie oft zum Auslesen der Sensoren verwendet, da sie sehr schnell ausgeführt wird

**CPU**

*Central Processing Unit*

der Hauptprozessor

**CSV-Datei**

*Comma-separated values*

Hierbei werden Messungen in einer Textdatei durch Zeilenumbrüche und einzelne Werte durch Beistriche getrennt

**D**

**Datenbus**

*System zur Datenübertragung zwischen mehreren Teilnehmern über einen gemeinsamen Übertragungsweg, bei dem die Teilnehmer nicht an der Datenübertragung zwischen anderen Teilnehmern beteiligt sind.*<sup>41</sup>

**E**

---

<sup>41</sup>Wikipedia, [Bus](#)— Wikipedia, Die freie Enzyklopädie.

## Glossar

### **Einplatinencomputer**

ein vollständiges Computersystem, welches auf einer einzelnen **Platine** zusammengefasst ist

### **F**

#### **Flickr**

ist eine Online-Plattform, auf der Fotos hochgeladen und veröffentlicht werden können

### **G**

#### **Gerätedatei**

eines der grundlegenden Prinzipien von diversen Linux-Betriebssystemen ist *Everything is a file*.

Daher können auf Festplatten, Schnittstellen und Informationen über das System einfach über das Auslesen von Dateien zugegriffen werden.

#### **Github**

*ein webbasierter Hosting-Dienst für Software-Entwicklungsprojekte*<sup>42</sup>

#### **Gnuplot**

ein Programm zur grafischen Darstellung von Messdaten und mathematischen Funktionen<sup>43</sup>

#### **GPIO**

*General Purpose Input/Output*

Kontakte auf der **Platine**, die softwareseitig für verschiedene Zwecke angesteuert werden können

z. B.: Auslesen von Sensoren, Ansteuern von Displays

### **H**

#### **Hertz**

die SI-Basiseinheit für die Frequenz

Sie gibt die Wiederholungen pro Sekunden an (hier: Wechsel zwischen *Strom* und *kein Strom* in der **CPU** pro Sekunde)

### **I**

#### **I<sup>2</sup>C**

*Inter-Integrated Circuit* (auf Deutsch gesprochen: *I-Quadrat-C*)

ein sehr weit verbreiteter **Datenbus**

### **J**

---

<sup>42</sup>Wikipedia, [GitHub — Wikipedia, Die freie Enzyklopädie](#).

<sup>43</sup>Wikipedia, [Gnuplot — Wikipedia, Die freie Enzyklopädie](#).

## Glossar

### JavaScript

eine Skriptsprache für dynamische Inhalte in Webseiten

### K

#### Kernelmodul

ein Programm, welches in das Betriebssystem geladen werden kann und oft zur Kommunikation mit Hardware verwendet wird

### L

#### LC-Display

*liquid crystal display*

Flüssigkristallbildschirm

#### LED

*light-emitting diode*

Licht abgebende Diode

#### Linux-Distribution

Es gibt nicht nur ein *Linux*, sondern eine sehr große Menge<sup>44</sup> Betriebssysteme, welche alle auf dem ursprünglichen *Linux*-Kernel basieren.

### O

### $\Omega$

Das Ohm ist die abgeleitete SI-Einheit des elektrischen Widerstands<sup>45</sup>

#### 1-Wire

ein Datenbus-System zur einfachen Kommunikation mit Sensoren

### P

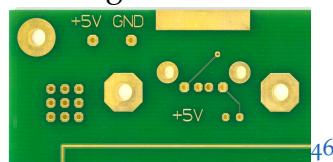
#### Pascal

ist die Einheit des (Luft-)Drucks

#### Platine

auch genannt Leiterplatte

ein Träger für elektrische Bauteile



<sup>44</sup>siehe diese Grafik: [de.wikipedia.org/wiki/Datei:Linux\\_Distribution\\_Timeline.svg](https://de.wikipedia.org/wiki/Datei:Linux_Distribution_Timeline.svg)

<sup>45</sup>Wikipedia, *Ohm — Wikipedia, Die freie Enzyklopädie*.

<sup>46</sup>Ulfbastel, *gold plated printed circuit board*.

## Glossar

### **Python**

ist eine 1991 entwickelte Programmiersprache, deren Fokus auf Programmlesbarkeit liegt.<sup>47,48</sup>

### **S**

#### **Shell**

ist eine Schnittstelle, über die der Benutzer Kommandos an den Computer schicken kann

#### **Standardabweichung**

ein Maß für die Streuung von Werten

#### **Steckbrett**

hierauf können schnell Schaltungen aufgebaut und getestet werden

#### **Streifenplatine**

eine [Platine](#), bei der die Kontakte streifenförmig miteinander verbunden sind.



<sup>49</sup>

### **V**

#### **VOC**

volatile organic compound (dt. Flüchtige organische Verbindungen)

#### **Volt**

die SI-Basiseinheit der elektrischen Spannung

---

<sup>47</sup>Python Software Foundation, [What is Python? Executive Summary](#).

<sup>48</sup>Rossum, [Python Reference Manual](#).

<sup>49</sup>Tfitzp, [Veroboard sample](#).

## Eidesstattliche Erklärung

Ich, Lukas Winkler, erkläre hiermit eidesstattlich, dass ich diese vorwissenschaftliche Arbeit selbständig und ohne Hilfe Dritter verfasst habe. Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als Zitate kenntlich gemacht und alle verwendeten Quellen angegeben habe.

Krems an der Donau, am \_\_\_\_\_  
Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_

## Zustimmung zur Aufstellung in der Schulbibliothek

Ich, Lukas Winkler, gebe mein Einverständnis, dass ein Exemplar meiner vorwissenschaftlichen Arbeit in der Schulbibliothek meiner Schule aufgestellt wird.

Krems an der Donau, am \_\_\_\_\_  
Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_

## Lizenz

Diese VWA und alle enthaltenen Bilder stehen, wenn nicht anders angegeben (alle Bilder im Glosar sind *Public Domain*), unter der *Creative-Commons-Lizenz Namensnennung - Weitergabe unter gleichen Bedingungen 4.0 International*.

Um eine Kopie dieser Lizenz zu sehen, besuchen Sie  
<http://creativecommons.org/licenses/by-sa/4.0/>.

