KellerBot Documentation

Release 0.1.0

s0556166 Steffen Exler

17.01.2019

Inhaltsverzeichnis

Warnung: Beta Software:

Diese Software hat noch nicht die stable Version erreicht. Es kann sein, dass sich die Struktur, Interface oder API ändern oder andere große Änderungen passieren, bevor die Version *1.0* erreicht wird.

- Online Dokumentation: https://kellerbot.readthedocs.io/de/latest/
- Quellcode: https://github.com/linuxluigi/kellerbot
- Präsentation: https://github.com/linuxluigi/kellerbot/raw/master/docs/_static/pr%C3%A4sentation.odp

`kellerBot` ist ein Raspberry Pi & Arduino Projekt, welches anhand eines 2 ardigen Kabels misst, ob Wasser an den Kabelenden ist, das Kabel kurzgeschlossen, das Kabel nicht angeschlossen oder ob das Kabel ohne geschlossenden Stromkreis angeschlossen ist. Diese Daten werden von einem Telegram Bot in einer Chat Gruppe angezeigt.

Die Abb. 1 ist das Projekt Logo, welches als Avatar für den Telegram Bot verwendet wird.

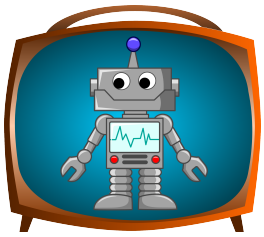

Abb. 1: Projekt Logo, Quelle: https://pixabay.com/en/android-bot-robot-television-happy-161184/

Hardware

Im der Abb. 2 ist der komplette Aufbau zu sehen.

- Raspberry Pi Modell B+ V1.2
- DHT22 Temperatur- & Luftfeuchtigkeitssensor
- mehrere Arduinos Nanos + jeweils 2 ardige Kupferkabel
- USB WLAN Stick oder *WLAN zu LAN Bridge*

Telegram Bot Befehle:

```
hilfe - zeige alle Befehle an
temperatur - Temperatur anzeigen
luftfeuchtigkeit - Luftfeuchtigkeit anzeigen
wassermelder - Wassertest
```

Abb. 2: Projekt Aufbau

KAPITEL 1

Theoretischer Aufbau

1.1 Zu lösendes Problem

Nachdem im letzten Jahr ein kleiner und größerer Wasserschaden im Keller bei uns aufkam und der große Wasserschaden in der Ferienzeit durch Zufall bemerkt wurde, überlegte ich mir, wie ein Warnmeldesystem aussehen könnte. Problematisch war bei dem großen Wasserschaden auch, dass dieser in der Ferienzeit auftrat und gut und gerne ein paar Tage unbemerkt hätte bleiben können. Damit wenn nun der Keller längere Zeit nicht mehr betreten wird, ein Wasserschaden rechtzeit erkannt wird, sollte nun ein Warnsystem installiert werden, für das ich auch die Erlaubnis der Hausverwaltung erhielt :)

1.2 Lösung

1.2.1 Grundkonzept

Im Keller konnte eine Stromversorgung über eine Steckdose sichergestellt werden und auch das WLAN Signal aus der Wohnetage ist im Keller noch ausreichend stark. Darum entschloss ich mich für eine einfache *Raspberry Pi* Lösung, bei der die erfassenten Sensor Daten über Telegram versendet werden sollten (Abb. 1.1).

Zusätzlich zur Wassermeldung sollten auch mittels eines *DHT22 Temperatur- und Luftfeuchtigkeitssensor* die Luftfeuchtigkeit und Temperatur auf Abfrage gemessen werden.

1.2.2 Aufbau 1: Raspberry Pi GPIO

Mein erster Ansatz war es über die *GPIO* Schnittstelle des *Raspberry Pi*'s zu messen, ob an den Kabelenden ein Stromkreislauf geschlossen wurde (in Abb. 1.2 Stromkreis schließen mittels Buttons dargestellt). Dafür war bis auf ein altes Telefonkabel und dem *Raspberry Pi* nichts weiter nötig, was den Versuch leicht umsetztbar machte.

Während der Umsetzung des Versuches sind mehrere Probleme aufgetreten:

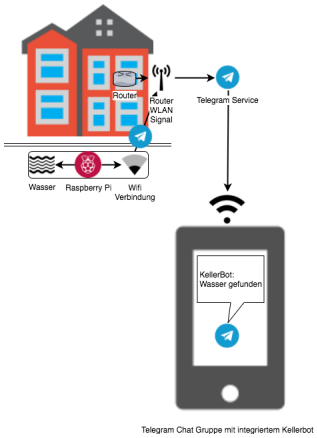

Abb. 1.1: Grundkonzept des zu lösenden Problems

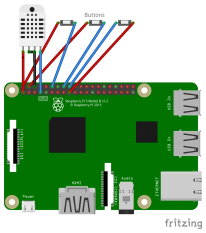

Abb. 1.2: Raspberry Pi *GPIO* Lösung

1. Es war in der Software nur möglich zu messen, ob ein Stromkreislauf geschlossen wurde oder nicht, es war nicht möglich festzustellen, ob das Kabel an den Enden kurzgeschlossen war oder ob überhaupt ein Kabel vorhanden war.
2. Die Messung erfolgte in zu großen Abständen, somit war die Aussagekraft nicht immer zuverlässig.
3. Es gab nur eine begrenzte Anzahl an Kabeln, die am *Raspberry Pi* angeschlossen werden konnten.

Der Code des Versuches kann im Branch feature/raspberry-pi-gpio-sensor-mode heruntergeladen werden.

1.2.3 Aufbau 2: Raspberry Pi und Arduino Nano

Um die in *Aufbau 1: Raspberry Pi GPIO* beschriebenden Probleme zu lösen, bot sich eine Lösung mit *Arduino Nanos* an, die an den Kabeln eine Kapazitätsmessung durchführen, womit sich mehrere Zustände auslesen lassen.

- kein Kabel an den Pin's angeschlossen
- Kabel ist kurzgeschlossen
- Kabel liegt im trockenen
- Kabel liegt im Wasser

In diesem Aufbau wird das Kabel fortlaufend auf diese Zustände geprüft und kann somit in Echtzeit die Daten zu Telegram senden.

Das dritte Problem kann durch einen aktiven USB Hub gelöst werden, der am *Raspberry Pi* angeschlossen wird. An dem Hub können eine große Zahl von Arduinos ausgelesen werden.

Messung der Kapazität über ein Arduino

Bei der Kapazitätsmessung wird geprüft, wie lange die Kapazität `C` benötigt um 63.2% ihrer gesamten Spannung zu laden. Dabei wird die Zeitkonstante `TC` über einen Widerstandskondensator `RC` innerhalb des Stromkreislaufs gemessen.

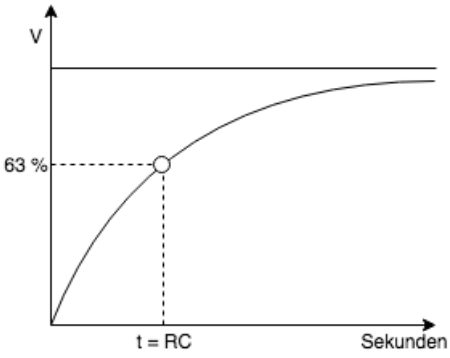

Abb. 1.3: Kapazitätsmessung

Größere Kapazitäten benötigen länger zum Laden. Deshalb erhalten diese eine größere Zeitkonstante. Die Kapazität in einer Widerstandskondensatorschaltung ist mit der Zeitkonstante durch folgende Formel verbunden:

$$Formel : TC = R \cdot C$$

- TC = Zeitkonstante in Sekunden
- R = Widerstand in Ohm
- C = Kapazität in Fahrad

Durch das Umstellung der Gleichung nach der Kapazität, ergibt sich folgende Gleichung:

$$C = \frac{TC}{R}$$

Nach den Messungen von http://www.circuitbasics.com/how-to-make-an-arduino-capacitance-meter/ kann der Arduino mit einer Schaltung mit nur 2 Drähten (Abb. 1.4 und Abb. 1.5) unbekannte Kapazitäten zwischen 470 uF und 18 pF messen.

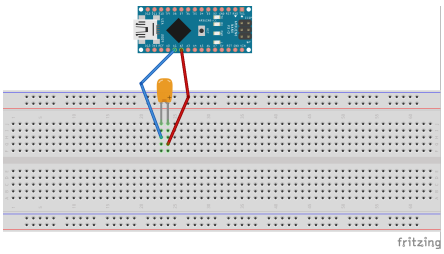

Abb. 1.4: Arduino Nano Schaltung

[1] [2]

1.3 Problem: Internet im Keller

Es besteht gibt kein direkte Netzwerkverbindung von der Wohnung bis zum Keller. Der WLAN Hotspot steht im 2.OG, wodurch bis zum Keller 3 Etage überbrück werden müssen. Um eine stabile Internetverbindung zu erhalten, gab es 2

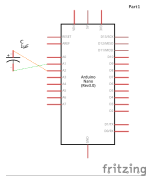

Abb. 1.5: *Arduino Nano* Schaltung schematische Darstellung

Lösungsmöglichkeiten, bei denen keine neue Hardware nötig war:

1.3.1 Powerline

Powerline ist ein Netzwerk über das Stromnetz, welches über mehrere Wohnungen verlegt werden kann. In meinem Test konnte habe ich Geräte von 2 verschiedenen Anbieter ausprobieren, wobei beide die Distanz gemeistert haben. Jedoch gab es auch ein erhöhtes Ausfallsrisiko, so dass es innerhalb einer Woche manuell neugestartet werden musste. Dies führte zum Ausscheiden dieser Möglichkeit.

1.3.2 W-LAN

Um herauszufinden, ob dieser Lösungsansatz möglich ist, schaute ich mir mittels der Android App Wifi Analyzer die Reichweite unseres 2.4 GHz WLAN's an und stellte fest, dass im Keller ein geringes aber stabiles Signal ankam.

Da ich zwischenzeitlich den USB WLAN Stick für den *Raspberry Pi* verloren habe, habe ich einen alten TP-LINK Router genommen und dort ein neues Betriebsystem openWrt aufgespielt. Somit konnte der WLAN Router nicht nur als *Access Point* dienen, sondern sich auch in ein anderes WLAN-Netzwerk einwählen und den Datenverkehr über Ethernet routen. Er konnte also als *WLAN zu LAN Bridge* arbeiten (Abb. 1.6).

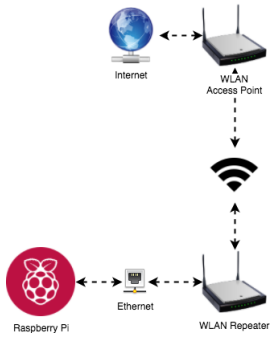

Abb. 1.6: *WLAN zu LAN Bridge* setup - Theorie

Dieses Setup sorgt nun auch dafür, dass sich bei Abbruch der WLAN Verbindung (z.B. durch einen Router Neustart des *Access Point*) der Brige Router von alleine wieder neu verbindet. Ein weiterer Vorteil dieser Methode gegenüber eines durchschinttlichen WLAN Sticks liegt in den Antennen des TP-Link Routers, welche sehr leistungsstark sind und sich gut in die Richtung des Signals ausrichten lassen (Abb. 1.7).

Abb. 1.7: WLAN zu LAN Bridge setup - Praxis

KAPITEL 2

Hardware

Für dieses Projekt wurde ein *Raspberry Pi Modell B+ V1.2*, 2 *Arduino Nano*, ein *DHT22 Temperatur- und Luftfeuchtigkeitssensor*, 2 *zwei ardriges nicht abgeschirmtes Kupferkabel* und ein *TP-LINK WLAN Router* verwendet.

2.1 Raspberry Pi

Um den *DHT22 Temperatur- und Luftfeuchtigkeitssensor* ansteuern zu können oder für den *Aufbau 1: Raspberry Pi GPIO* wird ein *Raspberry Pi* oder ein ähnlicher *Einplatinenrechner* mit *GPIO* benötigt.

2.2 DHT22 Temperatur- & Luftfeuchtigkeitssensor

Der *DHT22 Temperatur- und Luftfeuchtigkeitssensor* ist ein Sensor, der mit einer Platine ausgeliefert wurde und daher anders mit dem *Raspberry Pi* angeschlossen werden muss.

2.3 Arduino Nano

Für dieses Projekt wurden nicht Orginale *Arduino Nano* verwendet, sodurch die entsprechenden Treiber nachinstalliert werden müssen. Die Treiber sollten aber nur dann nachinstalliert werden, wenn diese nicht schon vom Werk aus auf dem System vorhanden sind, wie z.B. MacOs.

Auf der Rückseite des Nanos (Abb. 2.1) steht der Controller Name.

In diesen Fall ist der Treiber auf https://sparks.gogo.co.nz/ch340.html zu finden.

2.4 2 ardriges nicht abgeschirmtes Kupferkabel

Die Kabel werden für die Kapazitätsmessung benötigt. Mit Hilfe der Kapazitätsmessung ist es dann möglich, 4 verschiedene Zustände des Kabels zu messen, mitunter ob das Kabel unter Wasser ist. Wichtig bei dem Kabel ist es, dass

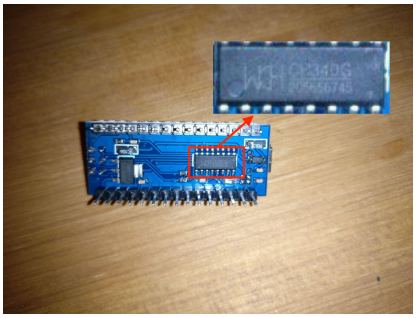

Abb. 2.1: *Arduino Nano* Controller Name

die beiden Kabelstränge direkt nebeneinander sind, wie bei ein Telefonkabel oder Lautsprecherkabel.

2.5 TP-LINK WLAN Router

Der *TP-LINK WLAN Router* ist optional, er dient ausschließlich als *WLAN zu LAN Bridge* um den *Raspberry Pi* über eine große Distanz zu dem *WLAN Access Point*. Dabei wird der Router mit dem quelloffenen Betriebsystem *openWrt* aufgespielt, womit er auch als *WLAN zu LAN Bridge* verwendet werden kann.

KAPITEL 3

Telegram

3.1 Über

Telegram ist ein dezentraler cloudbasierter Instant-Messenger, welcher ein offenes Protokoll und OpenSource Clients, verwendet. Außerdem ist es möglich eigene *Telegram Bot* zu programmieren, mit den die User interagieren können.

Offizielle Website: Telegram.org

3.2 Telegram Bot

Ein Telegram Bot ist ein Programm, welches auf jeder Plattform laufen kann und sich wie ein User im Chat mit extra Funktionen verhält. So ist es möglich, den Bot via Befehlen, die mit `/` starten zu steuern, wie zum Beispiel `/hilfe` welches die für diesen Bot möglichen Befehle anzeigt. Im FAQ auf der Offiziellen Telegram Seite wird der Telgram Bot genauer erklärt.

Für die Integration in diesen Projekt wurde der Python Telegram Wrapper python-telegram-bot verwendet, dieser erlaubt es mit wenigen Zeilen Code einen nützlichen Telegram Bot zu erstellen.

3.3 Telegram Bot erstellen

Info Diese Anleitung ist speziell für das Projekt `KellerBot` erstellt. Alle möglichen Funktionen des Telegram Bots sind auf Telegram Bot Entwickler Dokumentation nachzulesen.

3.3.1 1. BotFather einrichten

Der Telegram `BotFather` ist ein Telegam Bot, welcher zur Verwaltung der eigenen Telegram Bots dient. Mit diesem Bot ist es möglich, einen eigenen Bot zu erstellen, Befehle einzurichten, das Profilbild des Bots anzupassen und mehr.

Um Zugriff auf den `BotFather` zu erhalten, muss auf der Seite telegram.me/botfather auf `SEND MESSAGE` geklickt werden (Abb. 3.1).

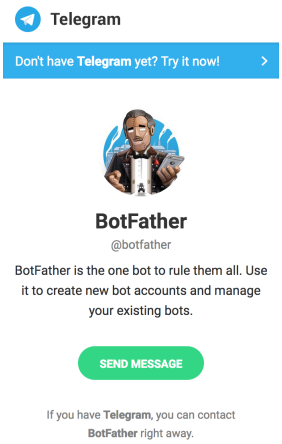

Abb. 3.1: BotFather hinzufügen

Anschließend öffnet sich das Telegram Fenster mit dem `BotFather` (Abb. 3.2). In diesem Fenster auf `START` klicken um dem `BotFather` zu interagieren.

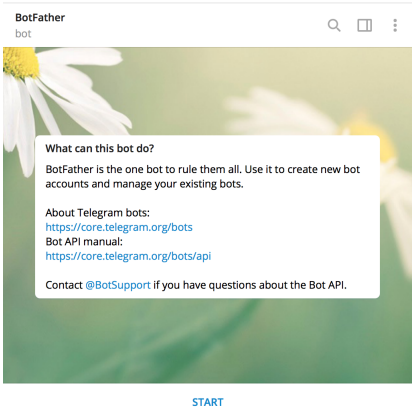

Abb. 3.2: BotFather Chat starten

Nachdem der Chat mit dem `BotFather` erstellt wurde, wird dieser im Kontaktverlauf angezeigt und kann so wieder direkt verwendet werden (Abb. 3.3).

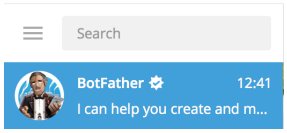

Abb. 3.3: BotFather Kontakt Verlauf

3.3.2 2. Eigenen Bot erstellen

Um einen neuen Bot zu erstellen, muss im Chat des `BotFather` (Abb. 3.3) der Befehl `/newbot` eingeben werden. Danach muss im Chat der Namen des neuen Bots geschrieben werden, in diesen Beispiel wird er `KellerBot` heißen.

Hinweis: Der Bot Name muss immer mit `bot` enden und muss einzigartig sein, also der Name darf noch nicht

vergeben sein! Beispiele `Keller_bot`, `KellerBot`, `kellerbot` oder `kellerBOT`.

Nachdem Erstellen des Bots gibt Telegram die URL & den *Token* des Bots aus:

> Done! Congratulations on your new bot. You will find it at t.me/KellerBot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.
>
> Use this token to access the HTTP API: 659931436:AAEAVoiIJxkswS-nl3tLBaTC1ydsgJn5SVA
>
> For a description of the Bot API, see this page: https://core.telegram.org/bots/api

Den soebend erstellten *Token* notieren. Dieser wird benötigt, damit das Programm später auf den `KellerBot` zugreifen kann.

Warnung: Der *Token* des Telegram Bots sollte nie veröffentlicht werden! Der Token in diesem Beispiel existiert auch nicht.

Um den gerade erstellten Bot in dem eigenen Chat zu aktivieren, muss auf den Bot Link geklickt werden. Der Link ist aus `t.me/` und deinem Botnamen zusammengesetzt. In diesem Beispiel heißt der Link `t.me/KellerBot`. Sobald auf den Link geklckt wurde, erscheint das Chat Fenster zum Bot. Dort muss wie bei dem `BotFather` Abb. 3.2 auf `START` geklickt werden muss, um den neuen Bot auf dem eigenen Account zu aktivieren.

3.3.3 3. Bot einrichten

Im `BotFather` Chat Fenster den gewünschten Keller Bot auswählen via `/mybots` und auf den neuen Bot klicken, siehe Abb. 3.4.

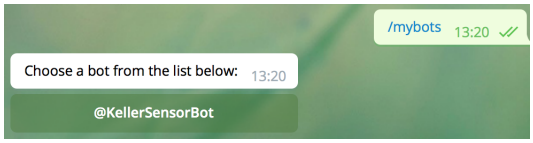

Abb. 3.4: Bot auswählen

In dem neuen Untermenü (Abb. 3.5) auf `Edit Bot` klicken.

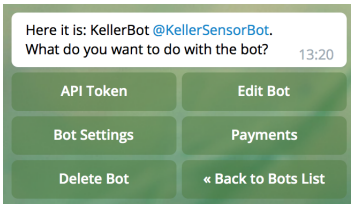

Abb. 3.5: Bot bearbeiten

Im Bot Optionsmenü (Abb. 3.6) können die Werte `Name`, `Beschreibung`, `About`, `Bild` und `Befehle` bearbeitet werden. Dafür auf die jeweilige Option klicken und danach den Wert im Chat eingeben oder im Fall des Bildes, das gewünschte Bild im Chat hochladen.

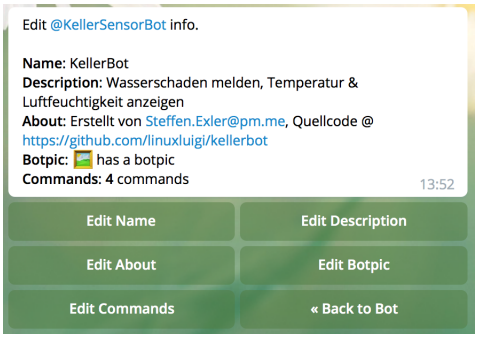

Abb. 3.6: Bot Optionsmenü

Standartwerte

Edit Name

```
KellerBot
```

Edit About

```
Erstellt von Steffen.Exler@pm.me, Quellcode @ https://github.com/linuxluigi/kellerbot
```

Edit Description

```
Wasserschaden melden, Temperatur & Luftfeuchtigkeit anzeigen
```

Edit Commands

```
hilfe - zeige alle Befehle an
temperatur - Temperatur anzeigen
luftfeuchtigkeit - Luftfeuchtigkeit anzeigen
wassermelder - Wassertest
```

Edit Botpic :

Bild von der online Doku android-image oder von der orginal Quelle https://pixabay.com/en/android-bot-robot-television-happy-161184/ herunterladen und anschließend im Chat als Bild einfügen (Abb. 3.7).

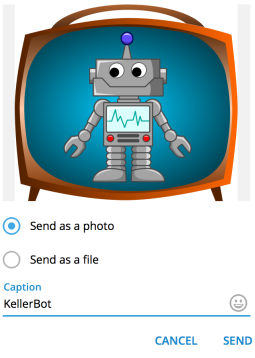

Abb. 3.7: Bot Bild einfügen

3.3.4 4. Telegram Chat Gruppe anlegen

Im Optionsmenü von Telegram auf `New Group` klicken, um dort eine Gruppe für den `KellerBot` zu erstellen (Abb. 3.8) z.B. mit den Namen `BasementWatchGroup`.

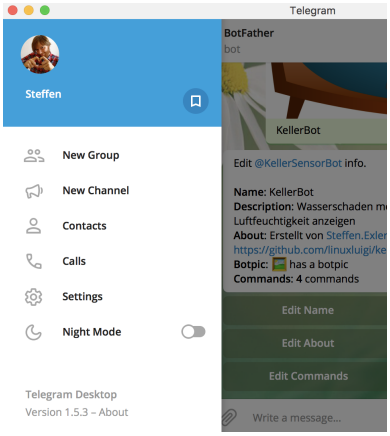

Abb. 3.8: Telegram Gruppe erstellen

Im nächsten Schritt erscheint das `Add Members` Fenster, hier den `KellerBot` suchen, hinzufügen (Abb. 3.9) und anschließend auf `Create` klicken, um die Gruppe zu erstellen.

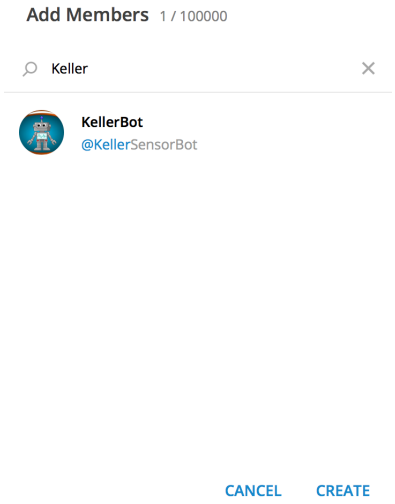

Abb. 3.9: Telegram Gruppe Mitglieder hinzufügen

3.3.5 5. Chat Group ID erhalten

Um die Chat Group ID zu erhalten, muss der `KellerBot` in der gewünschten Gruppe sein und es wird der *Token* benötigt (welcher in dem Kaptiel *2. Eigenen Bot erstellen* erstellt wurde). Außerdem muss noch mindestens eine Nachricht in der Gruppe geschrieben werden.

Die Chat Group ID ist via `https://api.telegram.org/bot<Token>/getUpdates` abrufbar. In diesem Beispiel würde die URL `https://api.telegram.org/bot659931436:AAEAVoiIJxkswS-nl3tLBaTC1ydsgJn5SVA/getUpdates` lauten. Die Ausgabe der URL ist ein JSON, wobei die Group ID unter `result -> 0 -> message -> chat -> id` zu finden ist

(Abb. 3.10). Diese ID sowie den *Token* zwischenspeichern. Diese Werte werden später bei der Einrichtung des Bots auf dem *Raspberry Pi* benötigt.

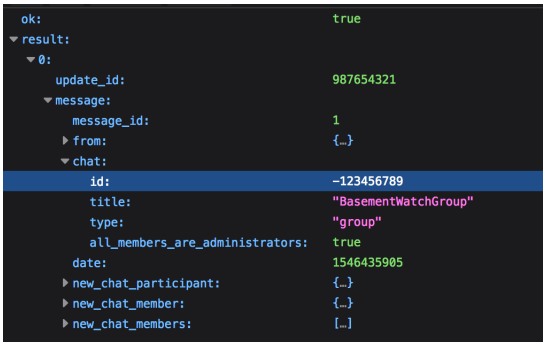

Abb. 3.10: Telegram Gruppe Mitglieder hinzufügen

KAPITEL 4

Installation

4.1 Vorbereitug

Für die Installation wird die *Hardware* und ein eingerichteter *Telegram* Bot benötigt.

4.2 Arduino

Den ersten Arduino Nano an einen Computer anschließen und die Arduino IDE starten (falls diese noch nicht installiert ist, kann diese via https://www.arduino.cc/en/Main/Software heruntergeladen werden).

Nach dem Starten den Inhalt der Datei `arduino-capacitance-meter/arduino-capacitance-meter.ino` in die Arduino Software einfügen.

Die IDE für den Arduino Nano einstellen

Arduino Nano: `Tools` -> `Board` -> `Arduino Nano`

Prozessor: `Tools` -> `Processor` -> `ATmega328P (Old Bootloader)`

Port: `Tools` -> `Port` -> `/dev/cu.wchusbserial14130` (der Port kann von System zu System anders aussehen)

Nachdem die Einstellungen gesetzt wurden kann

4.3 Raspberry Pi

4.3.1 DHT22 Temperatur- & Luftfeuchtigkeitssensor am Raspberry Pi anschließen

GPIO - Layout

Die Dokumentation wurde für das `Raspberry Pi Model B+ V1.2` erstellt. Wenn ein anderer Raspberry Pi verwendet wird, kann es sein das, dass das *GPIO* Layout anders aussieht und der *DHT22* Sensor an andere Pins

angeschlossen werden muss. Um das GPIO-Layout des Pi's herauszufinden, kann das Projekt https://github.com/RPi-Distro/python-gpiozero genutzt werden Abb. 4.1.

python-gpiozero installieren:

```
sudo apt install python3-gpiozero # install
pinout # run in cli
```

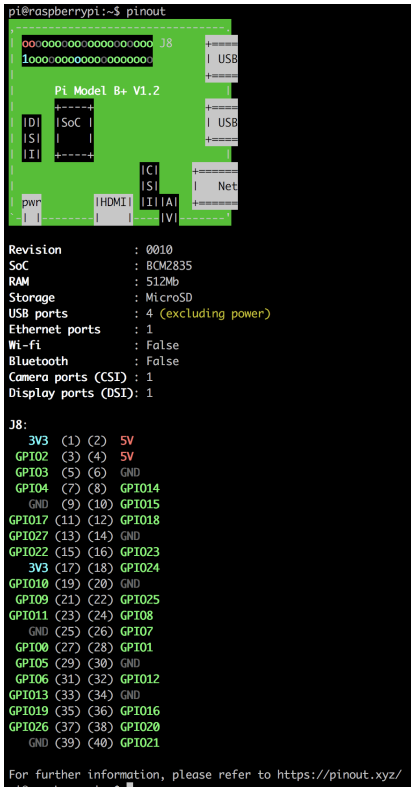

Abb. 4.1: Pinout

Beim Verwenden des `Raspberry Pi Model B+ V1.2` Models, die gleichen Pins verwenden, wie im Bild (Abb. 4.2) abgebildet.

Warnung: Bei dem `DHT22 Temperatursensor und Luftfeuchtigkeitssensor` im Bild handelt es sich um ein Modell, welches mit Platine ausgeliefert wurde. Der Steckplan ohne Platine weicht von dieser Abbildung ab!

DHT22 Temperatur- & Luftfeuchtigkeitssensor

- `+` -> `3V3`
- `out` -> `GPIO 7`
- `-` -> `GND`

4.3.2 Betriebsystem

Für dieses Projekt wird eine saubere Raspbian Installation vorausgesetzt, die aktuelle Version kann von der offiziellen Raspberry Pi Website heruntergeladen werden: https://www.raspberrypi.org/downloads/raspbian/

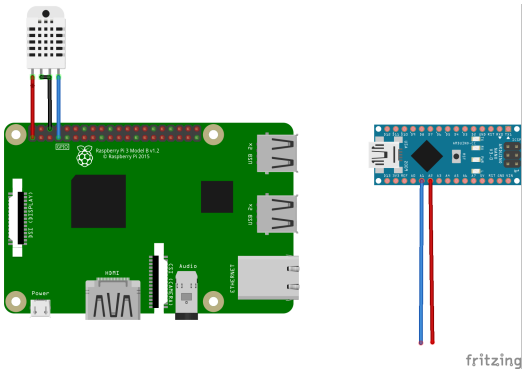

Abb. 4.2: Raspberry Pi Steckplan

4.3.3 KellerBot Software

KellerSensorTelegramBot kann auf dem Raspberry Pi über pip installiert werden:

```
$ sudo pip3 install git+git://github.com/adafruit/Adafruit_Python_DHT.git
$ sudo python3 -m pip install git+git://github.com/linuxluigi/kellerbot.git
```

Dieser Befehl lädt das Archiv und deren Abhänigkeiten aus dem Internet herunter und installiert diese.

Falls der Tarball heruntergeladen wurde, diesen entpacken und ausführen:

```
$ sudo pip3 install git+git://github.com/adafruit/Adafruit_Python_DHT.git
$ sudo python3 setup.py install
```

Nach der Installation des Pakets muss der Service geladen werden. Die Telegram Bot ID & Telegram Chat ID (für die Telegram ID's siehe die Kaptiel *5. Chat Group ID erhalten* & *2. Eigenen Bot erstellen*) in der service konfig Datei eintragen und anschließend den Service neuladen. Automatisches Starten während des Bootvorgangs aktivieren und den Service ausführen:

```
$ sudo systemctl daemon-reload
$ sudo systemctl edit keller.service

[Service]
Environment="BOT_ID=XXX"
Environment="CHAT_ID=XXX"

$ sudo systemctl daemon-reload
$ sudo systemctl enable keller.service
$ sudo systemctl start keller.service
$ sudo systemctl status keller.service
```

4.4 Keller Hardware Installation

Als erstes die Arduinos via USB an den Raspberry Pi anschließen, dann den Pi mit einem WLAN Stick oder via WLAN to Lan Bridge an das WLAN Netzwerk anschließen und alles mit Strom versorgen, wie in Abb. 4.3 dargestellt.

Die Kabel, die zur Messung dienen sollen, an einem Ende mit einem weiblichen Verbindungsstecker versehen und an die Pins `A2` & `A1` des jeweiligen Arduinos anstecken, wie in Abb. 4.4 zu sehen.

Abb. 4.3: Projekt Hardware Installation

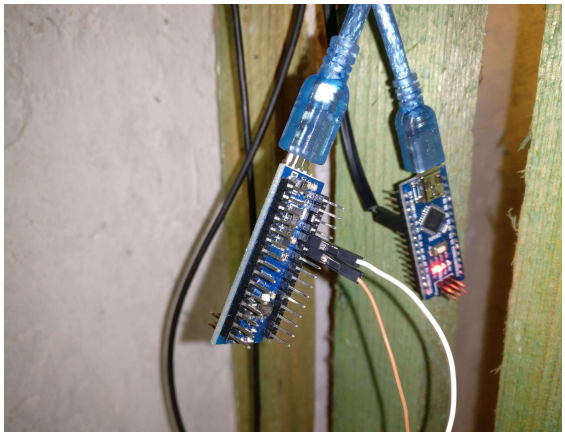

Abb. 4.4: Arduino Kabelanschluss

Im letzten Schritt die Kabel, die an die Arduinos angeschlossen wurden, im Keller verlegen. Mindestens das Ende muss am Boden liegen, um effektiv den Zustand messen zu können, wie in Abb. 4.5 und Abb. 4.6 zu sehen.

Abb. 4.5: Messkabel verlegen an der Decke

Abb. 4.6: Messkabelende verlegen

KAPITEL 5

Projekt Auswertung

5.1 Genauigkeit & Stabilität

In den Tests, in denen hauptsächlich die Kabelenden mit Wasser in Berührung kamen, funktionierten die Tests zuverlässig und innerhalb von 2 Sekunden wurden die Zustandsänderungen via Telegram gesendet. Es fehlte zum Schluss leider die Zeit für ausführlichere Tests. So wäre es interessant zu sehen, wie sich die Kapazität im Kabel ändert, wenn das Kabel zu Teilen in Wasser wäre, ohne dass die Kabelenden direkt mit Wasser in Kontakt kommen oder wie sich die Kapazität in den Kabeln ändert, wenn der Keller eine sehr hohe Luftfeuchtigkeit hat.

Der *Aufbau 1: Raspberry Pi GPIO* war über einen Monat im Keller in Betrieb. Bei diesem ersten Aufbau war die Netzwerk Anbindung über einen WLAN Stick geregelt. Dieser hatte zwar eine relativ starke Sende- & Empfangsleistung, aber die Reaktionszeit des *Raspberry Pi*'s war durch WLAN Abbrüche gemindert und dauerte des Öfteren über einige Minuten. Außerdem wurden im ersten Aufbau die Kabel nur alle 3 Minuten auf Wasser überprüft, so dass fließendes Wasser nicht immer als solches gemessen werden konnte. Trotz den Problemen mit dem Aufbau, war der *Raspberry Pi* die gesamte Zeit zuverlässig über den Telegram Bot erreichbar.

Der *Aufbau 2: Raspberry Pi und Arduino Nano* überzeugte hingegen mit der schnellen Reaktionszeit auf kurzzeitige Änderungen. Auch der Austausch des WLAN Sticks zu einen WLAN Router als *WLAN zu LAN Bridge* sorgte für eine schnelle Reaktionszeit des Telegram Bots. Der Aufbau steht nun seit zwei Wochen im Keller & arbeitet dabei fehlerfrei. Auch die Arduinos arbeiten im Dauerbetrieb, auch bislang fehlerfrei und senden innerhalb von Sekunden Zustandsänderungen an den *Raspberry Pi*.

5.2 Stromverbrauch

Der *Raspberry Pi* verbraucht ohne angeschlossene USB Geräte, mit Ethernet Anbindung und aktivem KellerBot Daemon 1,5 Watt pro Stunde. Mit jedem neu angeschlossenen *Arduino Nano* steigt der Verbrauch um 0,2 Watt die Stunde. So ergab es bei meinen Aufbau mit 2 angeschlossenden *Arduino Nanos* ein gesamt Verbrauch von 1,9 Watt pro Stunde, Stromverbauch ist auf Abb. 5.1 zu sehen. Bei durchschnittlichen 720 Stunden im Monat ergibt sich ein Monatsverbauch von 1,368 Kilowatt.

Abb. 5.1: Stromverbauch des Aufbaus mit 2 Arduinos, Anzeige in Watt pro Stunde

5.3 Fazit

Das angestrebte Ziel wurde kosteneffektiv gelöst, wobei nur 3 *Arduino Nanos* für ca 13€ nachgekauft werden mussten. Die restlichen Kabel & Hardware stammten aus alten Projekten und wurden nicht mehr verwendet. Auch können nun alle Bewohner des Hauses zeitnah einen Wasserschaden in Keller frühzeitig bemerken, wobei größere Schäden vermieden werden können.

Telegram überraschte mich bei der Arbeit äußert positiv. Einen eigenen Bot mittels Telegram zu erstellen stellte sich als sehr einfach dar. Auch ist es ein starker Vorteil, dass keine extra Software auf dem Smartphone installiert werden muss, um den KellerBot zu verwenden. Durch die leichte Bedienung ist Telegram auch ein praktisches Tool für Computer-Leihen: Sie müssen sich nicht erst an eine neue Software gewöhnen, sondern erhalten im Erstfall einfach zuverlässig eine Nachricht auf ihr Smartphone.

KAPITEL 6

Lizenz

KAPITEL 7

Glossar

Arduino Ein Open Source Hard- & Software Gerät mit einem einfachen Eingabe und Ausgabe Board, einem Mikrocontroller sowie analogen und digitalen Ein- und Ausgängen. Arduino soll eine einfache PLattform für Entwickler Einsteiger sein, die mit dieser Plattform programmieren und den Umgang mit Hardware lernen können.

Arduino Nanos

Arduino Nano Ist eine kompakte Version eines Arduinos.

DHT22 Temperatur- und Luftfeuchtigkeitssensor Der DHT22 ist ein Sensor zum Auslesen von Temperatur & Luftfeuchtigkeit, welcher leicht von einen Arduino, Raspberry Pi oder ähnlichen Geräten verwendet werden kann.

Einplatinenrechner Auf einer Leiterplatte befinden sich alle nötigen elektronischen Komponenten zum Betreiben eines Computersystems.

GPIO Abkürzung für *General Purpose Input Output*, dies bezeichnet programmierbare Ein- und Ausgänge für allgemeine Zwecke

openWrt Open Source Router Betriebsystem, welches meist mehr Funktionen bietet als die Betriebsysteme der Hardware Hersteller besitzen. OpenWart wird aktiv von Freifunkern verwendet um ein freies offenes Internet für alle anbieten zu können.

Powerline Ethernet über die Steckdose.

Raspberry Pi Ein Open Source Hard- & Software Minicomputer, der mit dem Ziel entwickelt wurde, nicht über 35$ zu kosten und sich durch geringen Stromverbrauch & durch Open Source Betriebsysteme für viele IT Projekte bewährt und beliebt gemacht hat.

Token Ein Schlüssel zur Authefizierung einer Schnittstelle.

WLAN Access Point

Access Point Basisstation zur kabellosen Kommunikation im LAN.

WLAN zu LAN Bridge Ein WLAN Router, welcher sich in ein exitierendes WLAN Netzwerk einwählt und dies als Internet Quelle dem LAN freigibt.

KAPITEL 8

Literaturverzeichnis

Literaturverzeichnis

[1] Circuit Basics \textbar Arduino \textbar 44. How to Make an Arduino Capacitance Meter. April 2015. URL: http://www.circuitbasics.com/how-to-make-an-arduino-capacitance-meter/.

[2] Arduino - CapacitanceMeter. URL: https://www.arduino.cc/en/Tutorial/CapacitanceMeter.

Stichwortverzeichnis