EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Quadrocopter Raspberry Pi 2016

Datenbank mit PythonMySQL

Universität Tübingen

Author:
Jascha Petter

Dienstag 20.08.16

Vorwort

Hier könnte ihr Vorwort stehen

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einrichtung

1.1 MySQL

Um auf ihrem System einen MySQL Server laufen zu lassen, benötigen Sie den „MySQL Community Server“.

1. Laden Sie sich hierzu den MySQL Community Server von der MySQL Homepage herunter.

   http://dev.mysql.com/downloads/mysql/[1]

   Wählen Sie dort die dementsprechende Version für Ihr System. Sie müssen sich für den Download nicht anmelden, oder ein Konto erstellen. Klicken Sie einfach auf „No thanks, just start my download“.

2. Nach der Installation wir Ihnen das Passwort für den Root User angezeigt.

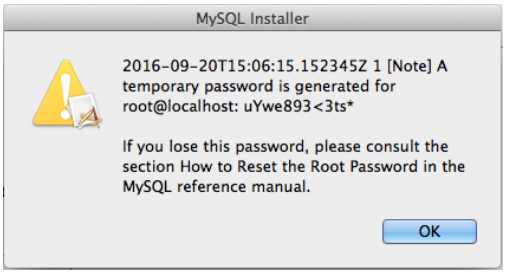

Abbildung 1.1: Passwort Alert

   Notieren Sie sich dieses, da es später noch gebraucht wird!

3. Öffen sie nun Ihren Termial und geben folgende Befehle ein, um ein neues Passwort zu vergeben:

```
$ cd /usr/local/mysql/bin
$ ./mysqladmin -u root -p password '<password>'
```

   Sie werden nun aufgefordert ihr Root Passwort, welches Sie vorhin notiert haben, einzugeben.

```
$ exit
```

   Um zurückzukehren

[1] stand 20.09.2016

4. Nun erstellen wir einen neuen Benutzer und geben diesem die Notwendigen Rechte.

```
$ ./mysql -u root -p
mysql > CREATE USER 'PiSense'@'localhost' IDENTIFIED BY 'somePW';
mysql > GRANT ALL PRIVILEGES ON *.* TO 'PiSense'@'localhost';
```

   Dies ist nun der neue Benutzer, mit dem gearbeitet wird.

5. Nun muss noch der MySQL-Server gestartet werden. Dies ist wie folgt möglich.

```
$ cd /usr/local/mysql/support-files/
$ sudo ./mysql.server  start
```

1.2 Datenbank

Da wir nun einen Benutzer zum arbeiten habe, fehlt nur noch die Datenbank und eine Tabelle.

1. Zu erst muss ein neues Schema erstellt werden:

```
CREATE DATABASE ‘SenseData‘ COLLATE 'latin1_swedish_ci';
```

2. Nun können wird eine Tabelle anlegen:

```
CREATE TABLE ‘SenseData‘.‘DATA‘ (
‘PITIME‘ timestamp(2) PRIMARY KEY NOT NULL,
‘ACC_X‘ double NOT NULL,
‘ACC_Y‘ double NOT NULL,
‘ACC_Z‘ double NOT NULL,
‘MAG_X‘ double NOT NULL,
‘MAG_Y‘ double NOT NULL,
‘MAG_Z‘ double NOT NULL,
‘G_ROLL‘ double NOT NULL,
‘G_PITCH‘ double NOT NULL,
‘G_YAW‘ double NOT NULL,
‘TEMP‘ double NOT NULL,
‘PRESS‘ double NOT NULL,
‘M1‘ double NOT NULL,
‘M2‘ double NOT NULL,
‘M3‘ double NOT NULL,
‘M4‘ double NOT NULL
) ENGINE='InnoDB' COLLATE 'latin1_swedish_ci';
```

   Zu den jeweiligen Einträgen später mehr.

Somit ist die Datenbank vollständig erstellt und kann mit Daten gefüllt werden.

1.3 Python

Um später Daten empfangen zu könne um diese in der Datenbank abzulegen benötigen wir Python. Da es sich um eine Skriptsprache handelt, ist eine IDE nicht vonnöten.

1. Laden sie sich hierzu Python 2.7 von der Python Homepage herunter.

   https://www.python.org/downloads/[2]

   Nach der Installation, können sie überprüfen ob es sich um die richtige version handelt mit:

   ```
   $ python --version
   ```

2. Damit Sie mit MySQL unter Python verwenden können, benötigen Sie noch MySQL-Python.

   https://pypi.python.org/pypi/MySQL-python/[3]

Nun ist alles komplett und Sie können beginnen.

[2] stand 20.09.2016
[3] stand 20.09.2016

2 Anwendung

Aufgabe des Python Skriptes ist es die Daten, welche vom Quadrocopter gemessen und per UDP versendet werden, entgegenzunehmen und in einer Datenbank abzulegen. Hierbei stellt das Skript eine Verbindung zu der Datenbank her und legt dort per INSERTs die empfangenen Daten ab.

2.1 Struktur der Datenbank

Spalte	Typ	Kommentar
PITIME	timestamp(2) [**CURRENT_TIMESTAMP(2)**]	
ACC_X	double	
ACC_Y	double	
ACC_Z	double	
MAG_X	double	
MAG_Y	double	
MAG_Z	double	
G_ROLL	double	
G_PITCH	double	
G_YAW	double	
TEMP	double	
PRESS	double	
M1	double	
M2	double	
M3	double	
M4	double	

Abbildung 2.1: Struktur der Datenbank

- **PITIME**
  PITIME ist die Zeit, welche momentan auf dem Pi herscht. Sie wurde als Primary Key gewählt, da es zu jeder Zeit nur einen Messwert geben kann. Angegeben wird sie als Timestamp mit einen Genauigkeit von 10ms.
- ACC_X/Y/Z
  Es werden alle drei Beschleunigungsachsen gespeichert, das Vorzeichen stellt hierbei die Richtung dar.
- MAG_X/Y/Z
  Für das Magnetfeld werden ebenfalls 3 Datensätze abgelegt.
- TEMP/PRESS
  Die Temperatur und der Luftdruck
- M1/2/3/4 Für die jeweiligen Motoren werden vier Datensätze abgelegt, da diese sich unabhänging von einander drehen können.

2.2 Auszug aus der Datenbank

Hier ist ein beispielhafter Auszug der Datenbank zu sehen, er zeigt einen Ausschnitt aus der Demo.

PITIME	ACC_X	ACC_Y	ACC_Z	MAG_X	MAG_Y	MAG_Z	G_ROLL	G_PITCH	G_YAW	TEMP	PRESS	M1	M2	M3	M4
2015-06-17 13:23:11.17	0.021556	-0.402376	-10.703669	0.000006	-0.000001	0.00002	-0.793481	0.534074	-0.80874	29.458334	983.614014	0	0	0	0
2015-06-17 13:23:11.27	0.079038	-0.388005	-10.639002	0.000006	-0.000001	0.00002	-0.915555	0.534074	-0.915555	29.460417	983.637695	1	1	1	1
2015-06-17 13:23:11.38	0.215558	-0.419141	-10.74678	0.000006	-0.000001	0.00002	0	0.259407	-0.686666	29.4625	983.619385	2	2	2	2
2015-06-17 13:23:11.49	0.045507	-0.3904	-10.564754	0.000006	-0.000001	0.00002	-0.656148	0.930815	-4.623554	29.4625	983.738037	3	3	3	3
2015-06-17 13:23:11.59	-0.028741	-0.562847	-10.447394	0.000006	-0.000001	0.00002	-0.473037	0.21363	-5.676443	29.466667	983.657471	4	4	4	4
2015-06-17 13:23:11.70	-0.941271	-0.079038	-11.151551	0.000006	-0.000001	0.00002	-0.335704	-0.259407	1.388592	29.466667	983.617676	5	5	5	5
2015-06-17 13:23:11.80	-1.712491	1.54723	-10.272552	0.000006	-0.000001	0.00002	-1.998962	0.80874	5.752739	29.464582	983.704834	6	6	6	6
2015-06-17 13:23:11.91	1.42508	-4.028546	-11.24017	0.000006	-0.000001	0.00002	0.564592	0.869778	5.73748	29.46875	983.663086	7	7	7	7
2015-06-17 13:23:12.12	1.369993	-0.744874	-9.673779	0.000006	-0.000001	0.00002	1.861629	3.418073	0.152593	29.46875	983.664551	8	8	8	8
2015-06-17 13:23:12.22	-1.207127	1.264609	-9.800719	0.000006	-0.000001	0.00002	-1.327555	-0.19837	1.144444	29.477083	983.758545	10	10	10	10
2015-06-17 13:23:12.33	-0.316152	-0.196398	-11.834152	0.000006	-0.000001	0.00002	0.106815	-0.167852	4.684591	29.477083	983.701172	11	11	11	11
2015-06-17 13:23:12.43	0.529315	-1.338857	-9.544444	0.000006	-0.000001	0.00002	-1.174963	0.595111	-1.724296	29.475	983.690918	12	12	12	12
2015-06-17 13:23:12.54	-2.062175	-1.952	-12.813745	0.000006	-0.000001	0.00002	-0.19837	-0.106815	-3.814814	29.477083	983.652832	13	13	13	13
2015-06-17 13:23:12.64	-1.92805	-3.408217	-10.22465	0.000006	-0.000001	0.00002	0.106815	-0.19837	0.61037	29.479166	983.716797	14	14	14	14
2015-06-17 13:23:12.75	1.362808	0.864628	-9.542049	0.000006	-0.000001	0.00002	-1.480148	0.167852	-0.19837	29.470833	983.660889	15	15	15	15
2015-06-17 13:23:12.85	0.174842	-0.184422	-11.529976	0.000006	-0.000001	0.00002	-1.02237	-0.122074	1.174963	29.466667	983.64502	16	16	16	16
2015-06-17 13:23:12.96	-0.90295	-2.18672	-10.976709	0.000006	-0.000001	0.00002	0.183111	0.045778	0.366222	29.466667	983.715576	17	17	17	17
2015-06-17 13:23:13.17	-1.046656	-0.464648	-9.822275	0.000006	-0.000001	0.00002	-0.701926	-1.052889	1.327555	29.458334	983.641602	19	19	19	19
2015-06-17 13:23:13.27	-0.203583	0.332918	-10.155192	0.000006	-0.000001	0.00002	-1.312296	-0.19837	1.571703	29.454166	983.624023	20	20	20	20

Abbildung 2.2: Daten der Demo

2.3 Python

2.3.1 Verbindung zur Datenbank

Um auf die vorhin erstelle Datenbank zuzugreifen, muss eine Verbindung hergestellt werden. Hierzu wird der neue Benutzer PiSense benötigt, natürlich könnten man auch Root benutzen das ist aber nicht Sinn der Sache.

```
import MySQLdb
mysql_server    = 'localhost' # SERVERNAME
mysql_user      = 'PiSense'   # USERNAME
mysql_pw        = 'somePW'    # PASSWORD
mysql_db        = 'SenseData' # NAME OF THE DATABASE

global db
try:
    db = MySQLdb.connect(mysql_server,mysql_user,mysql_pw,mysql_db)
except Exception as e:
    print("Connection to the DB failed!")
```

Abbildung 2.3: Connector

2.3.2 UDP Socket

Damit das Skipt eine Message vom Quadrocopter empfangen kann, muss ein UDP Socket auf einem bestimmten Port Lauschen. Python stellt uns hierbei einen Receive-Funktion zur verfügung. Es muss nur die IP und der Port and einen Socket gebindet werden.

```
UDP_IP = ""
UDP_PORT = 5000

sock = socket .socket(socket.AF_INET, # Internet(IPv4)
                      socket.SOCK_DGRAM) # UDP
sock.bind((UDP_IP, UDP_PORT))
```

Abbildung 2.4: UDP Socket

In diesem Fall lauschen wir auf dem Port 5000, benutzen IPv4 und natürlich UDP.

2.3.3 Empfangen und aufteilen der Daten

```
while True:
    data= sock.recv(1024) # buffer size is 1024 bytes
    if data[:4] == "AFAF":
        break
```

Abbildung 2.5: Empfange der Message

Es wird so lange gewartet bis eine Message ankommt, falls diese der String ÄFAFïst, wird das Skript beendet. Da die Daten in einem einzigen String ankommen, müssen sie herausgefiltert werden. Um dies zu erreichen wird der String bei jedem Zeilenumbruch getrennt. Nun müssen noch die Bezeichner *abgeschnitten* werden. Auch hier liefert uns Python einen Lösung mit. Es werden die Vier ersten Zeichen verworfen.

```
data = data.split("\n")
time_S= data[0][4:]
time_M= data[1][4:]
ACC_X = data[2][4:]
ACC_Y = data[3][4:]
ACC_Z = data[4][4:]
MAG_X = data[5][4:]
```

Abbildung 2.6: Trennen der Daten

2.3.4 Ablegen der Daten

Damit die empfangenen Date jetzt nicht verloren gehen, werden sie in die Datenbank geschrieben. Dies erfolgt über einen INSERT. Hierbei werden die Daten in ihre jeweiligen Spalten der Tabelle gespeichert. Am ende muss das Query noch commited werden an die Datenbank.

```
dbc.db.query("""
INSERT INTO `SenseData`.`DATA`
(`PITIME`,
`ACC_X`,`ACC_Y`,`ACC_Z`,
`MAG_X`,`MAG_Y`,`MAG_Z`,
`G_ROLL`,`G_PITCH`,`G_YAW`,
`TEMP`,`PRESS`,
`M1`,`M2`,`M3`,`M4`)
VALUES
(FROM_UNIXTIME("""+ts+"""),
"""+ACC_X+""","""+ACC_Y+""","""+ACC_Z+""",
"""+MAG_X+""","""+MAG_Y+""","""+MAG_Z+""",
"""+G_ROLL+""","""+G_PITCH+""","""+G_YAW+""",
"""+TEMP+""","""+PRESS+""",
"""+M1+""","""+M2+""","""+M3+""","""+M4+""");
""")
dbc.db.commit()
```

Abbildung 2.7: Query

Um sicherzustellen, dass das Skript funktioniert kann mit dem udpSend Skipt ein Senden des Quadrocopters simuliert werden. Jedoch handelt es sich hierbei nur um Zufallswerte.