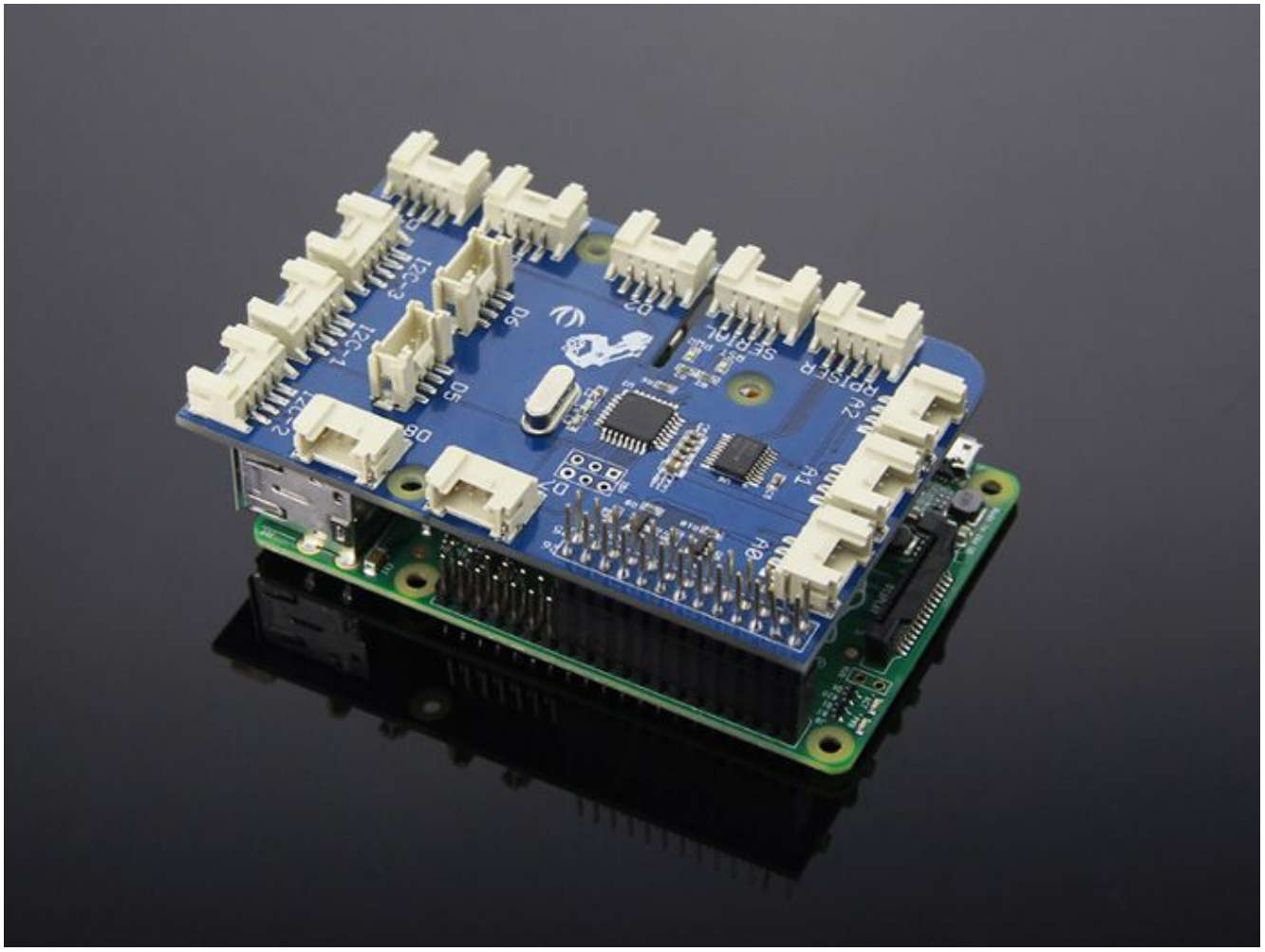
grovepi-home-logger - opis

Uruchomienie urządzenia

Raspberry Pi powinno mieć podłączoną do siebie nakładkę GrovePi, wyrównanie pinów GPIO jak na zdjęciu poglądowym:

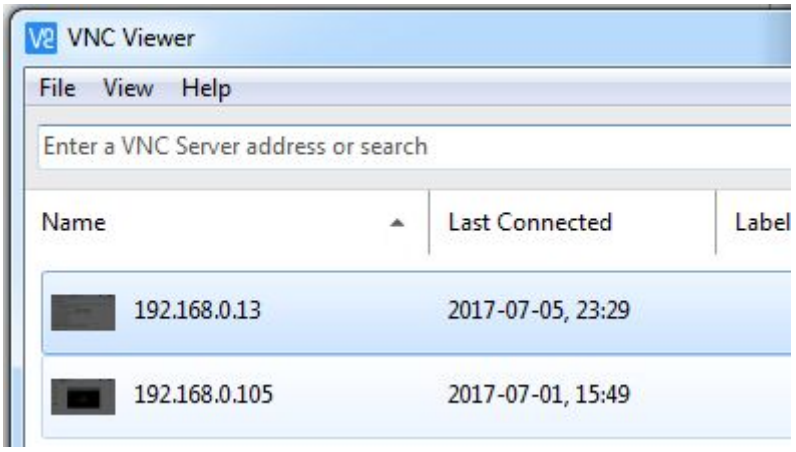


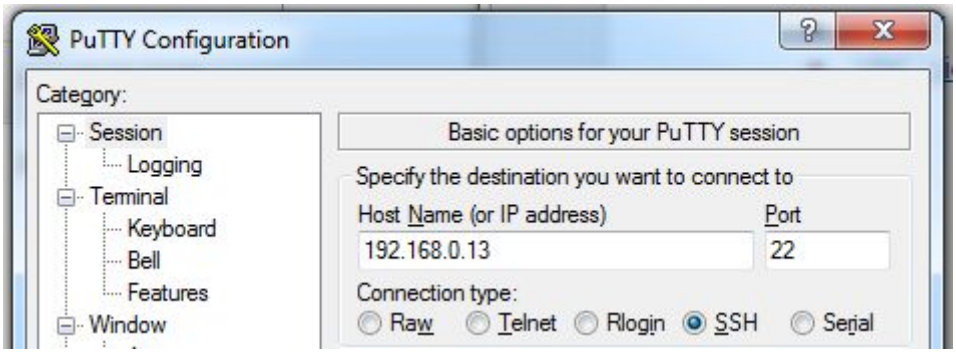
Podłączamy kabel microUSB do plytki (zielonej), kabel Ethernet do płytki i wolnego portu w routerze. Podłączamy kabel do gniazdka - powinny zaświecić się diody: **PWR i ACT** na Raspberry Pi i **PWR** na GrovePi. Jeśli tak, możemy przejść dalej.

Podłączenie do płytki

Możemy teraz podłączyć się do płytki przez dwa narzędzia:

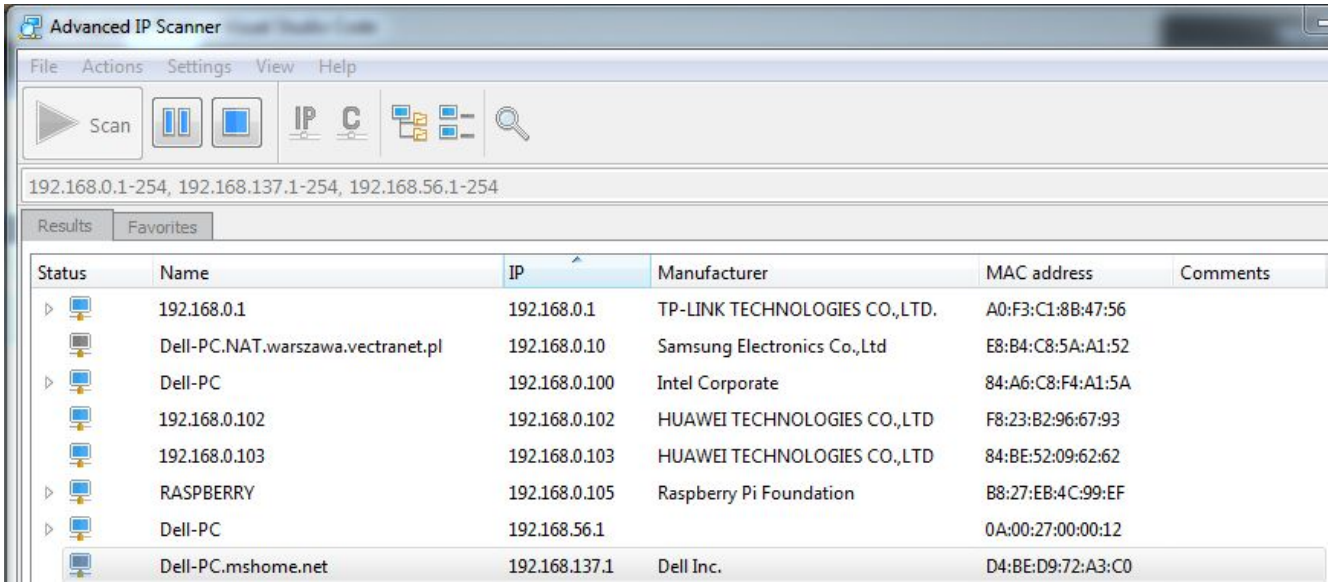
- PuTTY - tryb konsolowy,
- VNC Viewer - tryb graficzny,
- WinSCP - menadżer plików.





IP można znaleźć za pomocą programu Advanced Ip Scanner. Program ten służy do przeskanowania lokalnej sieci w poszukiwaniu urządzeń i nadanych im numerów IP.

- Name: **RASPBERRY**



Początkowe uruchomienie

Początkowy test uruchomienia jest dostępny pod tym adresem. To krótki test polegający na podłączeniu diody LED do płytki i sprawdzeniu, czy miga.

Opisy najważniejszych metod, funkcji, opis portów

- `pinMode()` ustawia tryb danego portu na `"INPUT"` - wejście, `"OUTPUT"` - wyjście.
- `analogRead(2)` i `digitalRead(2)` czytają z dwóch różnych portów na GrovePi.
    - `analogRead(2)` z portu oznaczonego A2,

    - `digitalRead(2)` z portu oznaczonego D2.
- Istnieją aliasy na porty analogowe (by użyć metod jak do portów cyfrowych):
    - A0 = D14
    - A1 = D15
    - A2 = D16,
- Stąd `analogRead(0)` i `analogRead(14)` zwrócą tę samą wartość.

A0, A1, A2 (D14, D15, D16)	analogowe	0-1023
D2-D8	cyfrowe, 1-bitowe	0-1
D3, D5, D6	analogowe, + PWM	0-255 (tylko zapis)

```
grovepi.analogRead(0)           - port A0, odczyt 0-1023
grovepi.analogRead(1)           - port A1, odczyt 0-1023
grovepi.analogRead(2)           - port A2, odczyt 0-1023
grovepi.analogRead(14)          - port A0, odczyt 0-1023
grovepi.analogRead(15)          - port A1, odczyt 0-1023
grovepi.analogRead(16)          - port A2, odczyt 0-1023
grovepi.analogWrite(3,val)          - port D3, zapis PWM 0-255
grovepi.analogWrite(5,val)          - port D5, zapis PWM 0-255
grovepi.analogWrite(6,val)          - port D6, zapis PWM 0-255
grovepi.digitalRead(2)          - port D2, odczyt 0-1
grovepi.digitalRead(3)          - port D3, odczyt 0-1
grovepi.digitalRead(4)          - port D4, odczyt 0-1
grovepi.digitalRead(5)          - port D5, odczyt 0-1
grovepi.digitalRead(6)          - port D6, odczyt 0-1
grovepi.digitalRead(7)          - port D7, odczyt 0-1
grovepi.digitalRead(8)          - port D8, odczyt 0-1
grovepi.digitalRead(14)         - port A0, odczyt 0-1
grovepi.digitalRead(15)         - port A1, odczyt 0-1
grovepi.digitalRead(16)         - port A2, odczyt 0-1
grovepi.digitalWrite(2,val)     - port D2, zapis 0-1
grovepi.digitalWrite(3,val)     - port D3, zapis 0-1
grovepi.digitalWrite(4,val)     - port D4, zapis 0-1
grovepi.digitalWrite(5,val)     - port D5, zapis 0-1
grovepi.digitalWrite(6,val)     - port D6, zapis 0-1
grovepi.digitalWrite(7,val)     - port D7, zapis 0-1
grovepi.digitalWrite(8,val)     - port D8, zapis 0-1
grovepi.digitalWrite(14,val)    - port A0, zapis 0-1
grovepi.digitalWrite(15,val)    - port A1, zapis 0-1
grovepi.digitalWrite(16,val)    - port A2, zapis 0-1
```

Aktualizacja firmware'u

Aktualizacja oprogramowania jest dostępna pod tym adresem.

Apache Derby

Apache Derby do ściągnięcia tutaj: http://db.apache.org/derby/derby_downloads.html

Zweryfikować instalację Apache Derby można następującą komendą:

- **`java org.apache.derby.tools.sysinfo`**

```
pi@raspberry:/opt/Apache/db-derby-10.13.1.1-bin/bin $ java org.apache.derby.tools.sys
------------------ Java Information ------------------
Java Version:    1.8.0_65
Java Vendor:     Oracle Corporation
Java home:       /usr/lib/jvm/jdk-8-oracle-arm32-vfp-hflt/jre
Java classpath:  /opt/Apache/db-derby-10.13.1.1-bin/lib/derby.jar:/opt/Apache/db-derb
/lib/derbyoptionaltools.jar:
OS name:         Linux
OS architecture: arm
OS version:      4.9.24-v7+
Java user name:  pi
Java user home:  /home/pi
Java user dir:   /opt/Apache/db-derby-10.13.1.1-bin/bin
java.specification.name: Java Platform API Specification
java.specification.version: 1.8
java.runtime.version: 1.8.0_65-b17
--------- Derby Information --------
[/opt/Apache/db-derby-10.13.1.1-bin/lib/derby.jar] 10.13.1.1 - (1765088)
[/opt/Apache/db-derby-10.13.1.1-bin/lib/derbytools.jar] 10.13.1.1 - (1765088)
[/opt/Apache/db-derby-10.13.1.1-bin/lib/derbyoptionaltools.jar] 10.13.1.1 - (1765088)
```

Żeby zapewnić, że zmienne środowiskowe dostępne są z poziomu konsoli i połączenia ssh, można dodać je do plików `~/.bashrc` i `~/.bash_profile`.

Baza danych

Dostęp do IJ (SQL), komenda: **`java org.apache.derby.tools.ij`**
Stworzenie bazy danych: **`connect 'jdbc:derby:/usr/db/<db_name>;create=true';`**
Podłączenie do bazy danych: **`connect 'jdbc:derby:/usr/db/<db_name>';`**
Uruchomienie skryptu - stworzenie tabel: **`run 'init.sql'`** (po podłączeniu do bazy).

Struktura tabeli SENSOR_INFO:

Nazwa kolumny	**Typ**
ID	ID INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMENT BY 1)
LOG_TIME	TIMESTAMP
TEMPERATURE	REAL
HUMIDITY	REAL
LIGHT	REAL
SOUND	REAL
PROXIMITY	REAL
THRESHOLD	INTEGER

```
ij> run 'C:\Users\Grzegorz\Desktop\init.sql';
ij> CREATE TABLE SENSOR_INFO (
        ID INTEGER NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMEN
T BY 1),
        LOG_TIME TIMESTAMP,
        TEMPERATURE REAL,
        HUMIDITY REAL,
        LIGHT REAL,
        SOUND REAL,
        PROXIMITY REAL,
        THRESHOLD INTEGER);
0 rows inserted/updated/deleted
ij> ALTER TABLE SENSOR_INFO ADD PRIMARY KEY (ID);
0 rows inserted/updated/deleted
ij>
```

```
ij> insert into SENSOR_INFO (temperature, humidity) values (2.50, 4.50);
1 row inserted/updated/deleted
ij> select * from SENSOR_INFO;
ID          |LOG_TIME                   |TEMPERATURE    |HUMIDITY       |LIGHT
          |SOUND          |PROXIMITY      |THRESHOLD
------------------------------------------------------------------------------
------------------------------------------------
1           |NULL                       |2.5            |4.5            |NULL
          |NULL           |NULL           |NULL

1 row selected
ij>
```

Przygotowane skrypty

Logger mieszkaniowy - Logger

Prosty logger, którego można użyć do logowania informacji z mieszkania pod naszą nieobecność. Korzystamy z następujących komponentów:

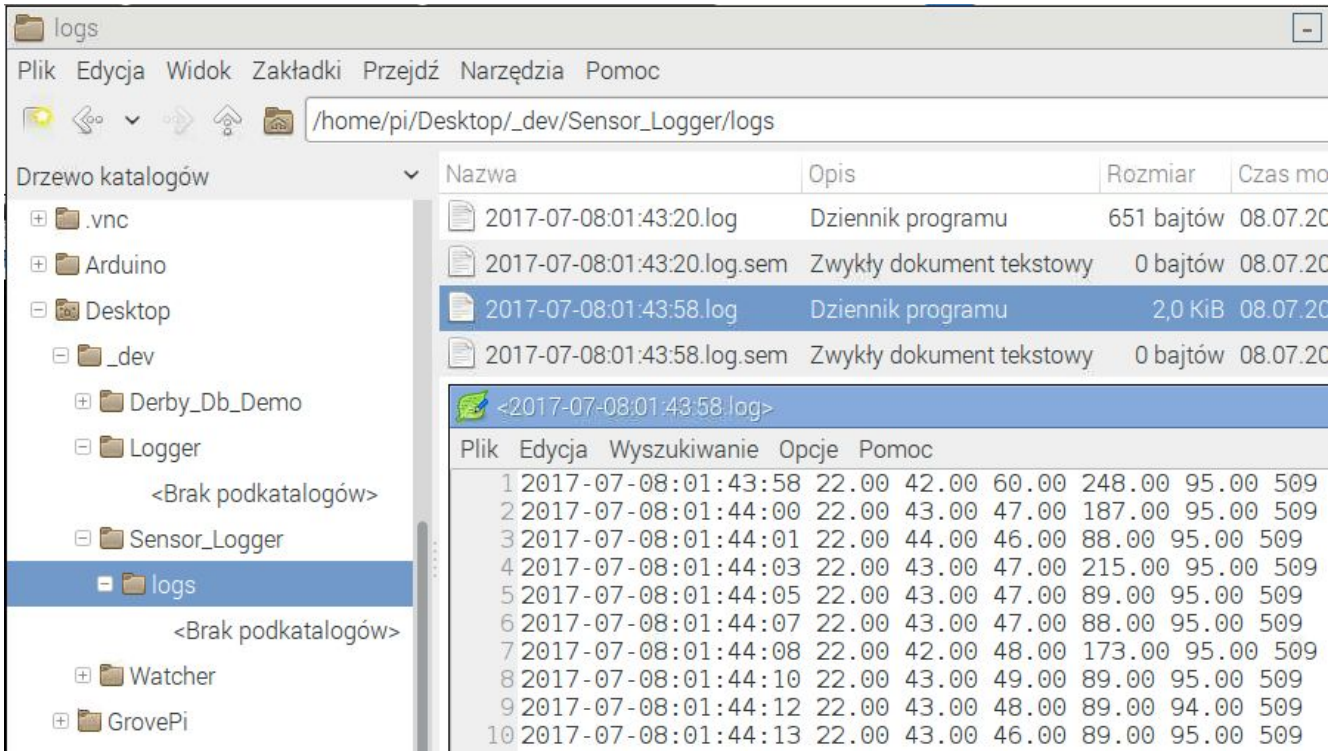
- wyświetlacz LCD RGB, port I2C-1,
- czujnik światła, port A0,
- czujnik dźwięku, port A1,
- potencjometr, port A2,
- czujnik temperatury i wilgotności, port D2,
- dioda LED, port D3,
- odległościomierz, port D4,
- dioda LED, port D5,
- przycisk, port D6,
- brzęczyk, port D7.

Funkcjonalności:

- zapis danych z czujników do pliku tekstowego (folder /logs),
    - nazwa pliku: **%y%m%d_%H%M%S.log**
    - format linii: %s **%.2f %.2f %.2f %.2f %.2f %d**
        - timestamp, temperatura, wilgotność, światło, dźwięk, odległość, threshold.
- zapis odbywa się co jedną sekundę,
- przyciśnięcie przycisku przez dłużej niż sekundę powoduje zakończenie działania,
- informacja dot. obecnie obliczonej odległości przedstawiona jest na wyświetlaczu,
- ekran wyświetlacza podświetlony jest kolorem, którego poszczególne składowe to:
    - R - odległość,
    - G - poziom światła,
    - B - poziom dźwięku,
- druga linia na ekranie zajęta jest przez wartość dot. alarmu dźwiękowego,
- jeśli odległość jest większa niż ustawiona wartość alarmu, alarm podniesie się.

Ponadto, skrypt po zakończeniu działania (po przyciśnięciu przycisku) wystawia plik semafora służący drugiemu skryptowi (tzw. watcher'owi) uruchomienie skryptu (loadera) służącego do pobrania danych z pliku tekstowego i wstawienie ich do bazy danych. **Sugerowanym sposobem na zakończenie logowania jest przyciśnięcie przycisku (port D6). Wtedy tworzy się semafor.**

Format pliku tekstowego:



Watcher

Watcher to prosty skrypt napisany w Bashu, który w pętli nasłuchuje na pojawienie się pliku o rozszerzeniu .sem w danym folderze. Po wykryciu takiego pliku, usuwa on semafor i przekazuje nazwę pliku do skryptu loadera, by wstawić rekordy do bazy danych. Parametry:

- nasłuchiwanie na pliki o nazwie: **<nazwa_pliku>.txt.sem**,
- nasłuchiwanie działa w pętli nieskończonej,
- **ścieżka do folderu jest pierwszym argumentem skryptu**,
- uruchomienie:
  `./watcher.sh <folder_z_logami>`

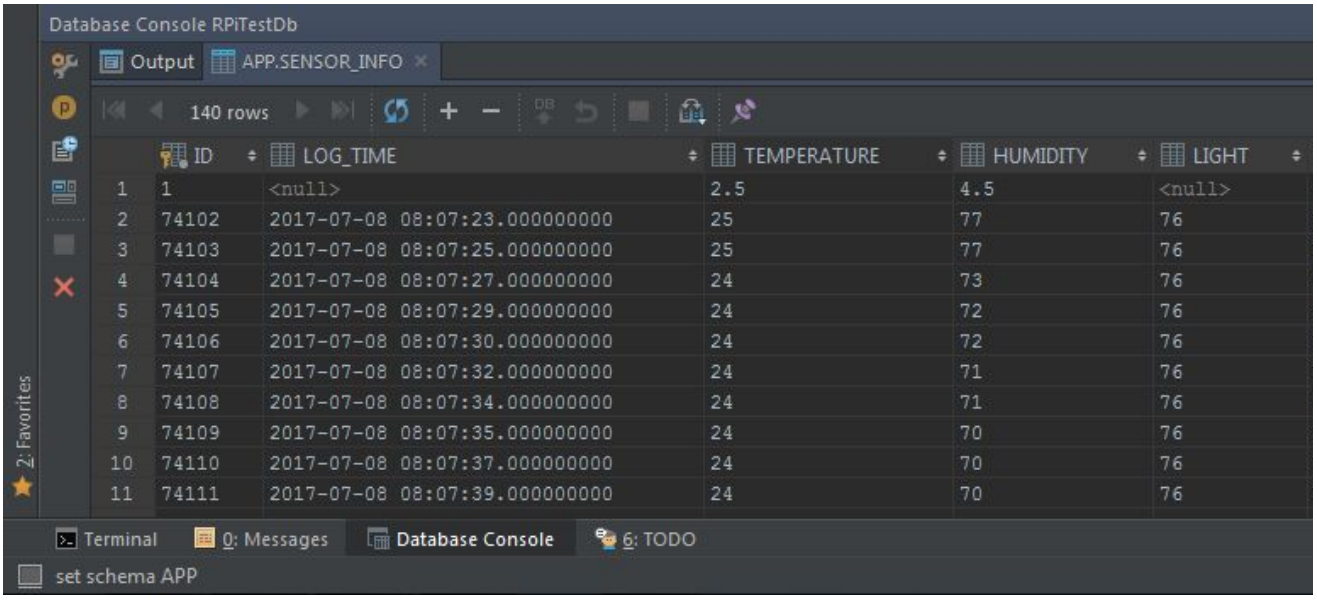
Plik znajduje się w lokalizacji: **/home/pi/Desktop/_dev/Watcher.**

Loader

Loader to program napisany w Javie, który służy do pobrania danych z pliku tekstowego. Nazwa tego pliku jest pierwszym argumentem pliku tekstowego. Służy ono do wczytania danych do bazy Apache Derby. **Aby program poprawnie się wykonał, nie może istnieć połączenie do bazy z poziomu ij.** Łączymy się do Derby w trybie embedded, nie klient - serwer, więc musimy mieć połączenie na wyłączność. Wyjście z ij: `exit;`.
Jeżeli wciąż instancja bazy jest uruchomiona, należy ją zamknąć komendą:

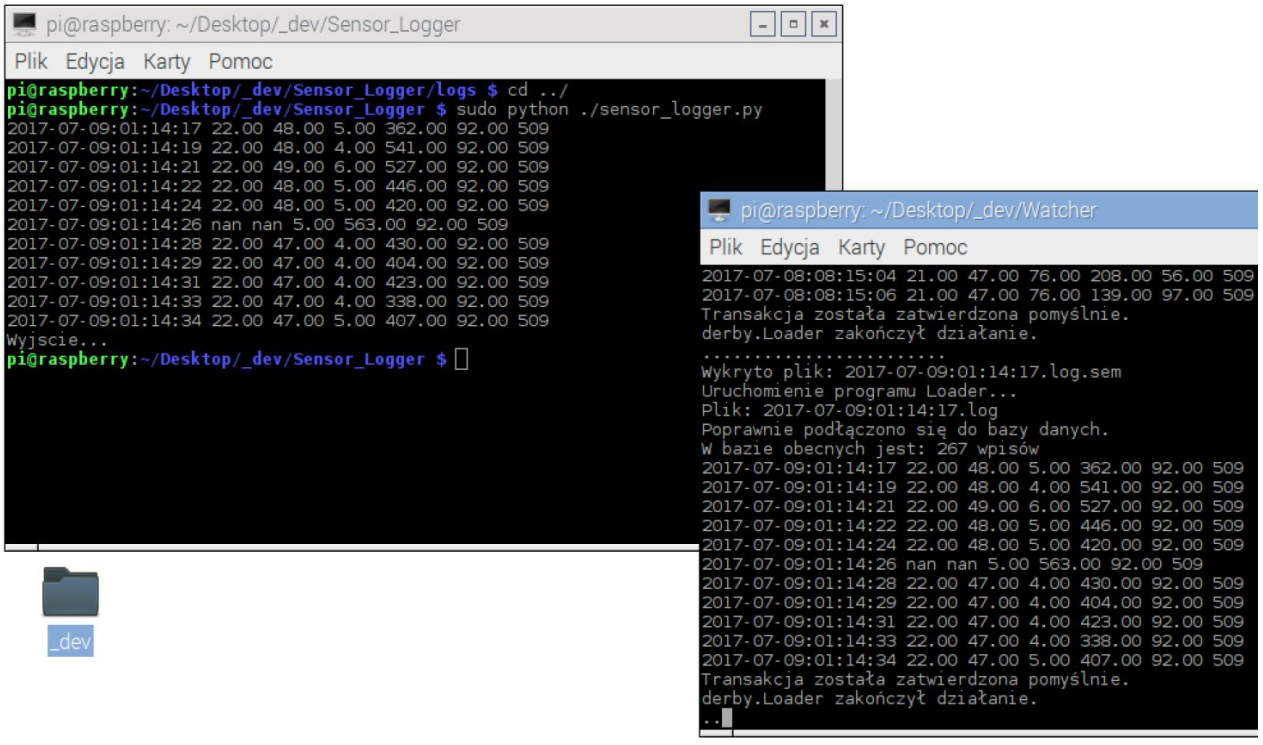
```
C:\Users\Grzegorz>java org.apache.derby.tools.ij
ij version 10.13
ij> connect 'jdbc:derby:;shutdown=true';
ERROR XJ015: Derby system shutdown.
ij>
```

Jeżeli wciąż występuje błąd z połączeniem, można spróbować usunąć wszystkie procesy, które korzystają z Derby i jednocześnie usunąć wszystkie pliki **.lck** z folderu bazy danych.



	ID	LOG_TIME	TEMPERATURE	HUMIDITY	LIGHT
1	1	<null>	2.5	4.5	<null>
2	74102	2017-07-08 08:07:23.000000000	25	77	76
3	74103	2017-07-08 08:07:25.000000000	25	77	76
4	74104	2017-07-08 08:07:27.000000000	24	73	76
5	74105	2017-07-08 08:07:29.000000000	24	72	76
6	74106	2017-07-08 08:07:30.000000000	24	72	76
7	74107	2017-07-08 08:07:32.000000000	24	71	76
8	74108	2017-07-08 08:07:34.000000000	24	71	76
9	74109	2017-07-08 08:07:35.000000000	24	70	76
10	74110	2017-07-08 08:07:37.000000000	24	70	76
11	74111	2017-07-08 08:07:39.000000000	24	70	76

Uruchomienie programu: **`java -jar Loader-1.0-SNAPSHOT.jar <nazwa_pliku>.log`**



W jaki sposób uruchomić skrypty?

1. W tle **uruchamiamy** *Watcher*:
   `./watcher.sh ../Logger/logs`
2. **Uruchamiamy** w drugim terminalu *Logger*:
   `sudo python ~/Desktop/_dev/SensorLogger/logger.py`
3. **Kończymy** *Logger* - przyciskiem,
4. Wykrycie semafora, uruchomienie *Loadera (automatycznie),*

5. *Loader* importuje dane do bazy danych,
6. **Wracamy** do punktu 2.