

Environ.Me

The smart University protection

Kai Hähre, Jens Erdmann, Timo Gerken, Rüdiger Bartz, Thomas Fischer,
Nicolas Albers

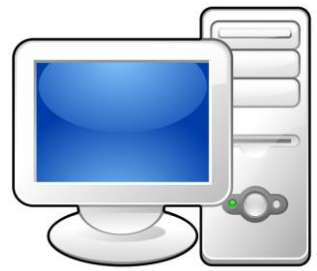
Agenda

- ▶ 1. Hintergrund
- ▶ 2. Aufbau
- ▶ 3. Sensoren
 - 1. Temperatur
 - 2. Luftfeuchtigkeit
 - 3. Raumhelligkeit
 - 4. Co - Messung
 - 5. IO -LED-Control
- ▶ 4. CoAP-Server
- ▶ 5. Webserver
- ▶ 6. Produkt Präsentation

Hintergrund

- ▶ Smart University mit RIOT Projekt
- ▶ Projektname: Environ.Me
 1. **Environ(.)ment** - (dt. Umwelt) Messung der Umgebungstemperatur, Luftfeuchtigkeit, Raumhelligkeit und des CO-Gehaltes
 2. Environ Me - (dt. Umgebe mich) Leitet im Brandfall Feuerwehrmänner\ -frauen und Zivilisten auf den schnellsten Weg nach draußen.

Aufbau



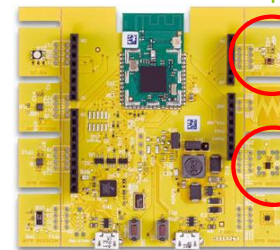
HTTP

RaspberryPI



CoAP

Phytec - PhyWave



Sensoren

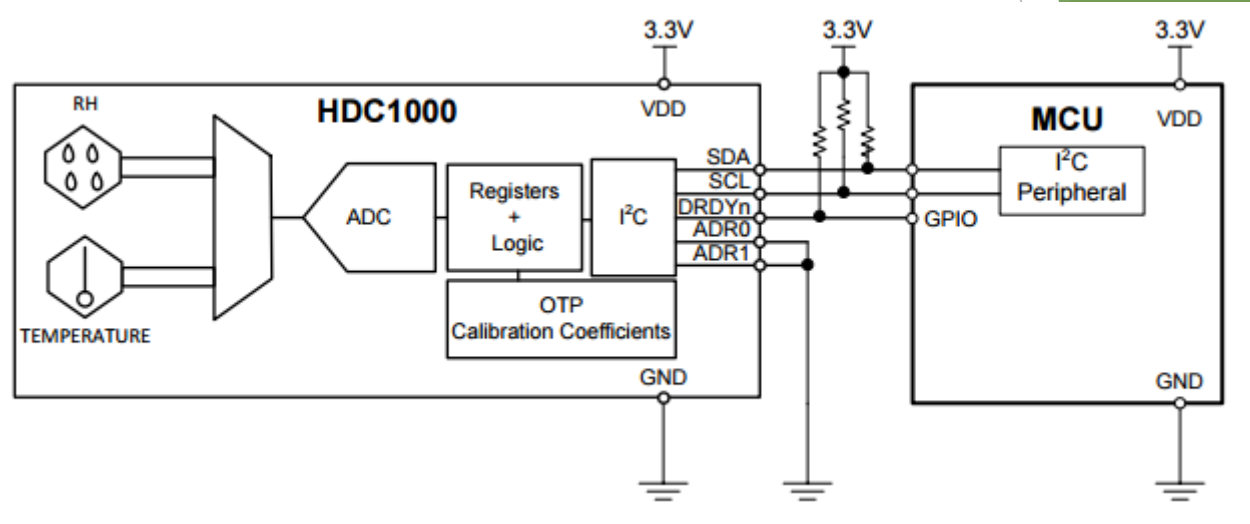


Sensoren

- ▶ Temperatur
- ▶ Luftfeuchtigkeit
- ▶ Raumhelligkeit
- ▶ CO-Messung
- ▶ IO - LED-Control

Temperatur

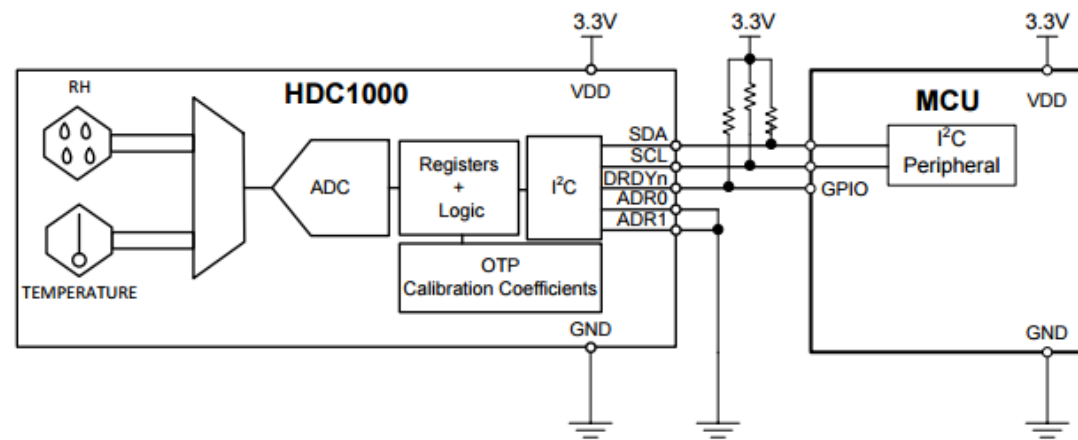
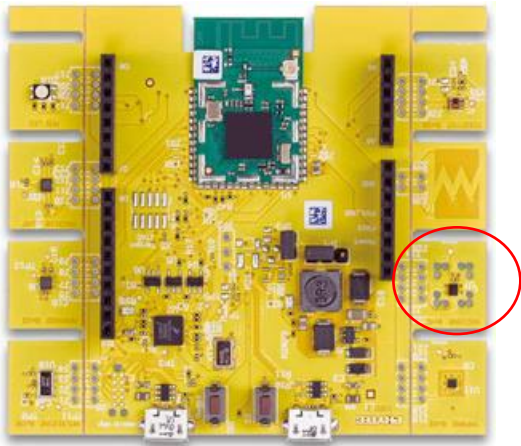
- Sensor HDC1000



```
if (hdc1000_startmeasure(&devHdc)) {  
    puts("HDC1000 Start measure failed.");  
    return NULL;  
}  
xtimer_usleep(HDC1000_CONVERSION_TIME); //26000us  
  
hdc1000_read(&devHdc, &rawtemp, &rawhum);  
printf("HDC1000 Raw data T: %5i  RH: %5i\n", rawtemp, rawhum);  
  
hdc1000_convert(rawtemp, rawhum, &temp, &hum);  
printf("HDC1000 Data T: %d  RH: %d\n\n", temp, hum);  
xtimer_sleep(1);
```

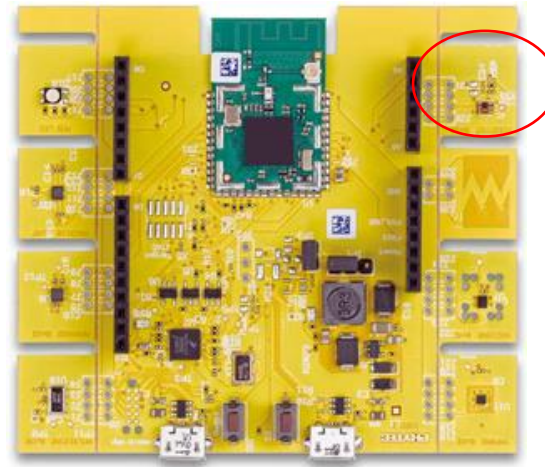
Luftfeuchtigkeit

- Sensor HDC1000
- Implementiert wie Temperatursensor



Raumhelligkeit

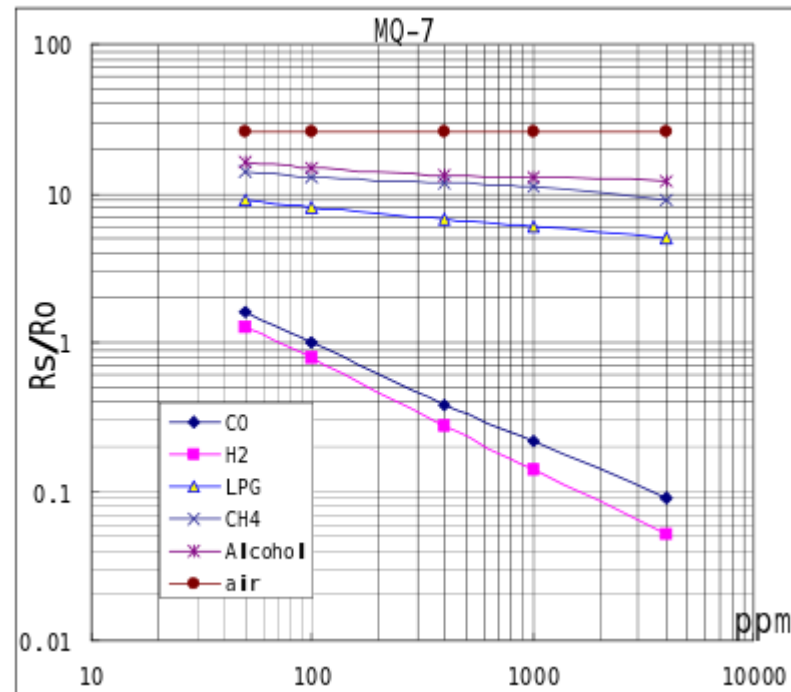
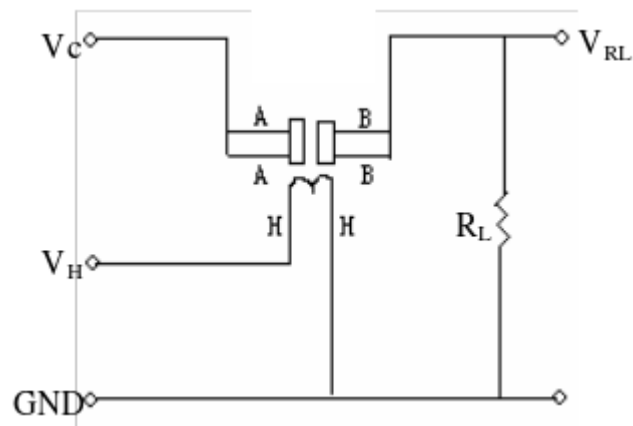
- Sensor TCS37727
- RGB Lichtsensor
- Photodiode mit Farbfilter



```
if (tcs37727_set_rgbc_active(&dev)) {
    puts("Measurement start failed.");
    return NULL;
}
tcs37727_read(&dev, &data);
printf("R: %5"PRIu32" G: %5"PRIu32" B: %5"PRIu32" C: %5"PRIu32"\n",
data.red, data.green, data.blue, data.clear);
printf("CT : %5"PRIu32" Lux: %6"PRIu32" AGAIN: %2d ATIME %d\n",
data.ct, data.lux, dev.again, dev.atime_us);

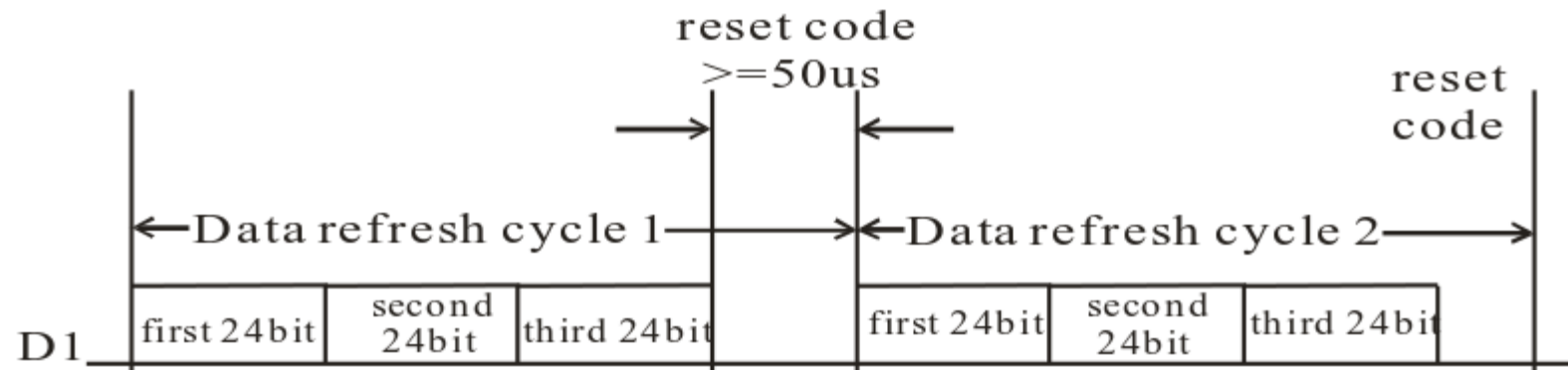
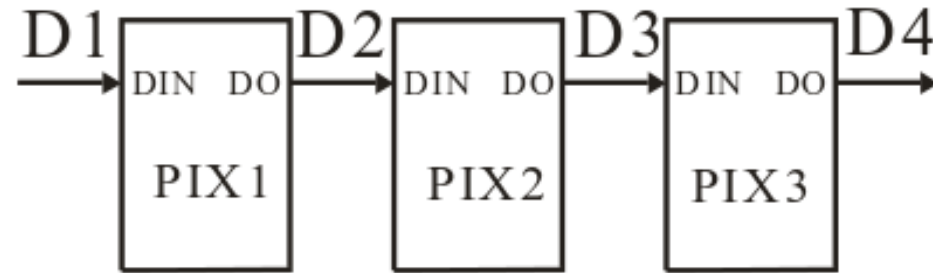
xtimer_usleep(SLEEP);
```


Co-Messung

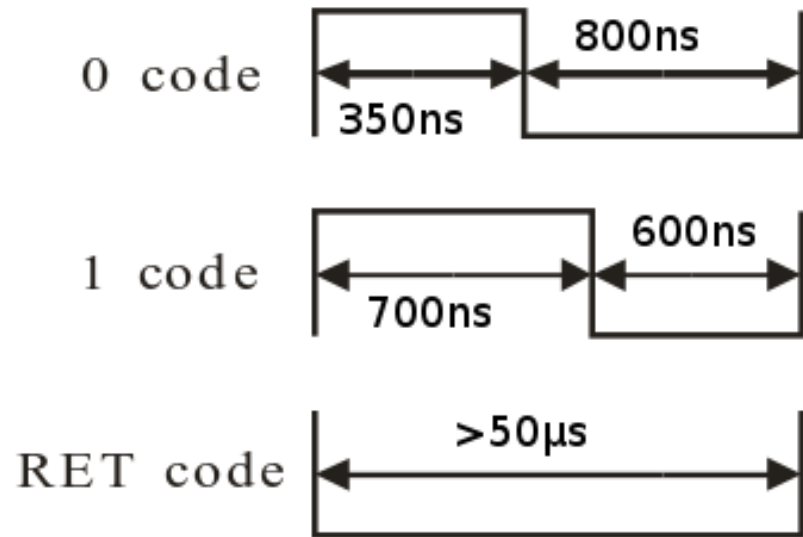


```
1 #include "coSensor.h"
2 #include <math.h>
3
4 float adcCoCalc(int value)
5 {
6     return log((94.37751004f/value)-0.06084945844f)*1900/(-1.8971f)+100;
7 }
```

IO-LED-Control



IO-LED-Control



```
void sendZero(void)
{
    for(int i=0;i<3;i++)
    {
        LED_R_OFF;
    }

    for(int i=0;i<10;i++)
    {
        LED_R_ON;
    }
}
```

Starten des Servers

- Starten der Threads:

- Led Control

```
void start_led_control(void) {  
    led_control_pid = thread_create(thread_stack, sizeof(thread_stack),  
                                    THREAD_PRIORITY_MAIN, THREAD_CREATE_STACKTEST,  
                                    led_control, NULL, "led_control");  
}
```

- CoAP-Server

```
void start_server(void) {  
    thread_create(thread_stack, sizeof(thread_stack), THREAD_PRIORITY_MAIN,  
                 THREAD_CREATE_STACKTEST, server, NULL, "coap_server");  
}
```

Ansteuerung der Sensoren

- Initialisierung der Sensoren

```
| int init_sensors(void)
```

- Sensor HDC1000:
 - Temperatur
 - Luftfeuchtigkeit
- Sensor TCS37727
 - Beleuchtungsstärke

Ansteuerung der Sensoren

- Messwertabfrage mittels Get Request
- HDC1000

```
int get_temperature(void)
```

Gibt Temperaturwert *100 zurück

```
int get_humidity(void)
```

Gibt Luftfeuchtigkeitswert *100 zurück

Ansteuerung der Sensoren

- Messwertabfrage mittels Get Request
- TCS37727

```
long get_illuminance(void)
```

Gibt die Beleuchtungsstärke zurück

Ansteuerung der Sensoren

- Messwertabfrage mittels Get Request
- Abfrage aller Messwerte

```
void get_all(int *temp, int *hum, long *lux)
```


Ansteuerung des LED-Bands

- Läuft im eigenen Thread
- Ansteuerung über Put Request

0 : LED-Band aus

1 : LED-Band leuchtet von rechts nach links

2 : LED-Band leuchtet von links nach rechts

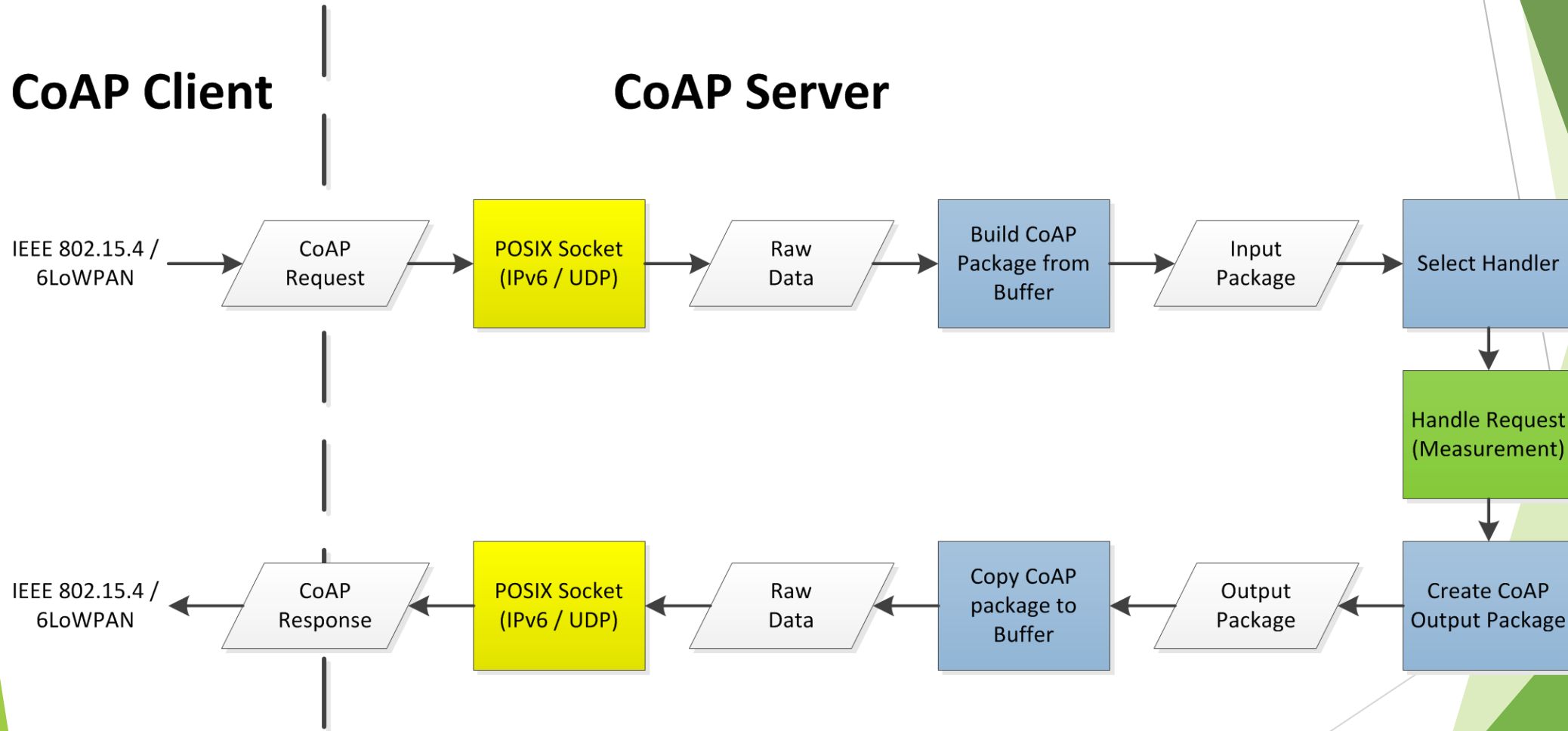
Ansteuerung des LED-Bands

- Kommunikation mit CoAP-Server
- msg Struct Reference
- Messages zwischen den Threads
- Zwischenspeicherung als msg Queue

CoAP-Server

- Server läuft in einem eigenen Thread
- RIOT POSIX Sockets + microcoap
- GET Requests für die Messungen
- PUT Request für das LED Band

CoAP Request Handling



CoAP-Client

- Mehrere Ansätze
 - Python (mit aiocoap)
 - C (mit libcoap)
 - Bibliothek
 - Beispiel Client wurde für Mysql angepasst

CoAP-Client

- Libcoap
 - Schwierigkeiten
 - Kommunikation zum Phyttec
 - Einbinden der libcoap Bibliothek
 - Libcoap und MySQL

CoAP-Client

- LED-Steuerung

```
pi@raspberrypi ~/libcoap/examples $ echo 2 | coap-client -m put coap://[fe80::f8e3:4e62:71ba:600a%lowpan0]/led -f-v:1 t:CON c:PUT i:e58a {} [ ]
```

- Datenbank

```
mysql> SELECT * FROM temperatures;
```

id	sensor_id	dt	temperature
1	1	2016-01-14 16:35	2779
2	1	2016-01-14 16:35	2779
3	1	2016-01-14 16:35	2780
4	1	2016-01-14 16:35	2780
5	1	2016-01-14 16:35	2779


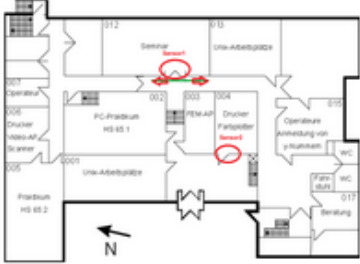
Webserver

- Raspberry Pi
- MySQL-Datenbank
- Python + Flask + html

[Home](#)

Environ.me!

The date and time on the server is: 2016-01-15 08:04



Sensors

ID	ipv6	Position	LED-Status	LEDs off(0)	LEDs left(1)	LEDs right(2)
1	fe80::5bb3:4e48:6f6c:6002	R07.61 BT7	0	<input type="button" value="LEDs off"/>	<input type="button" value="LEDs left"/>	<input type="button" value="LEDs right"/>
2	fe80::f3a3:4e62:71ba:600a	Flur 7.Stock BT7	0	<input type="button" value="LEDs off"/>	<input type="button" value="LEDs left"/>	<input type="button" value="LEDs right"/>

[Details](#)

Latest Temperatures

Sensor ID	Datum+Zeit	Temperature
1	2016-01-14 00:32	28.45°C
2	2016-01-14 00:32	25.17°C

Maximum Temperatures

Sensor ID	Datum+Zeit	Temperature
1	2016-01-13 23:28	35.37°C
2	2016-01-14 00:32	25.17°C

MySQL-Database

- Database: Environme

```
show fields in temperatures;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
sensor_id	int(11)	YES	MUL	NULL	
dt	varchar(18)	YES		NULL	
temperature	int(11)	YES		NULL	





```
show fields in sensors;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
ipv6	varchar(50)	YES		NULL	
Position	varchar(50)	YES		NULL	
led	int(11)	YES		NULL	

```
show fields in humidity;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
sensor_id	int(11)	YES	MUL	NULL	
dt	varchar(18)	YES		NULL	
humidity	int(11)	YES		NULL	

Webserver

-  Environme
- run.py
-  app
- __init__.py
- views.py
-  templates
- base.html
- index.html
- details.html
-  static
- raumplan2.png
- logo-large.png

views.py

```
@app.route('/')
@app.route('/index')
def index():
    now = datetime.datetime.now()
    timeString = now.strftime("%Y-
```

```
    curs.execute ("SELECT sensor_id, dt, temperature FROM temperatures WHERE sensor_id=%s AND temperature = (SELECT max(temperature) FROM temperatures WHERE sensor_id=%s)
    maxt2 = curs.fetchall()
```

```
    curs.close()
    db.close()
```

```
    return render_template('index.html',
        title = 'Environ.me!',
        time = timeString,
        rows = rows,
        values1 = values1,
        values2 = values2,
        maxt1 = maxt1,
        maxt2 = maxt2)
```

Ausschnitt aus 'index.html'

```
{% for row in rows %}
    <tr>
        <td><b>{{ row[0] }}</b></td>
        <td>{{ row[1] }}</td>
        <td>{{ row[2] }}</td>
        <td>{{ row[3] }}</td>
```

Produkt Präsentation

Vielen Dank für Ihre
Aufmerksamkeit.

Habt Ihr noch Fragen?