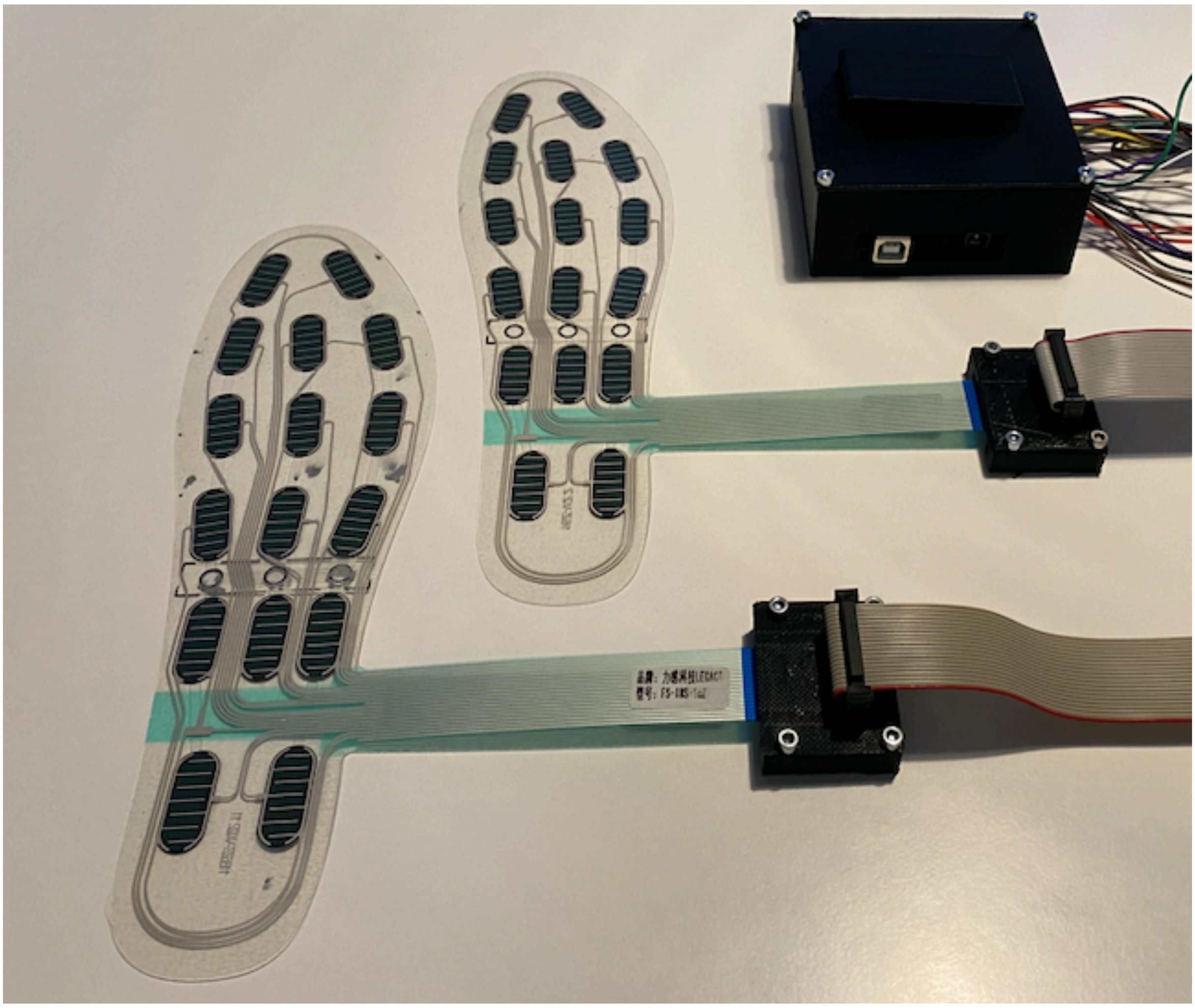
Project: Smart insoles

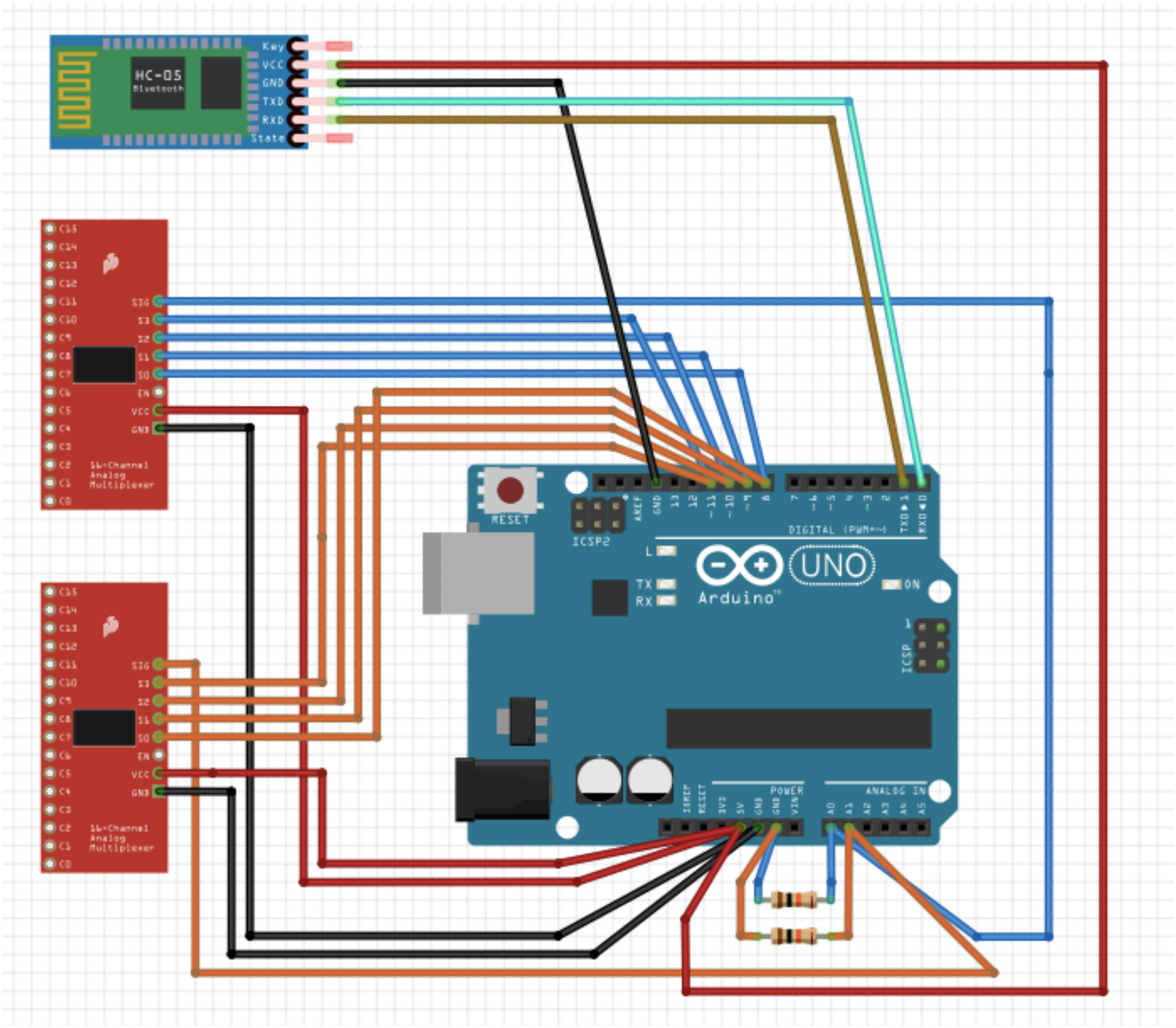
The program allows to record food pressure by during walking and running. After the record, the data can evaluate with this program.



Hardware:

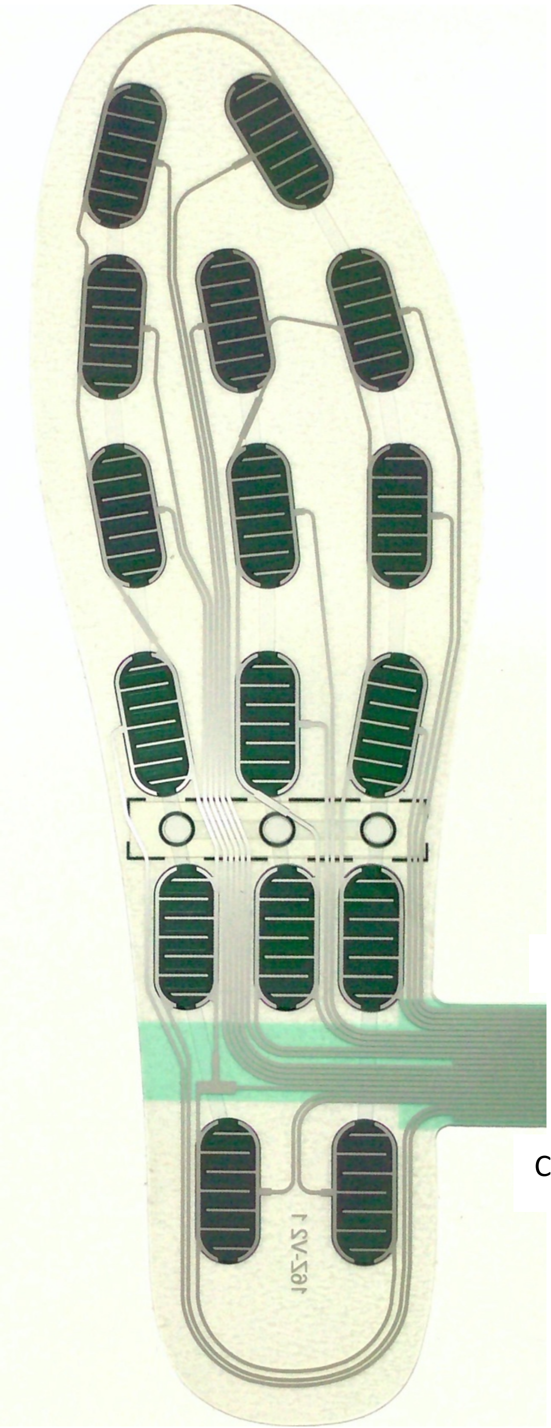
1 x Arduino Uno
2 x FSR food sensor plates
2 x FCP connectors 1.0mm
2 x 74HC HCT151 16-input multiplexer
2 Meter flat cable (20 pin)
1 x Bluetooth module HC-05
1x Battery adapter for the power supply of the Arduino
Some cable and adapters for the connections

Wiring of the multiplexer and the Bluetooth module with the Arduino:

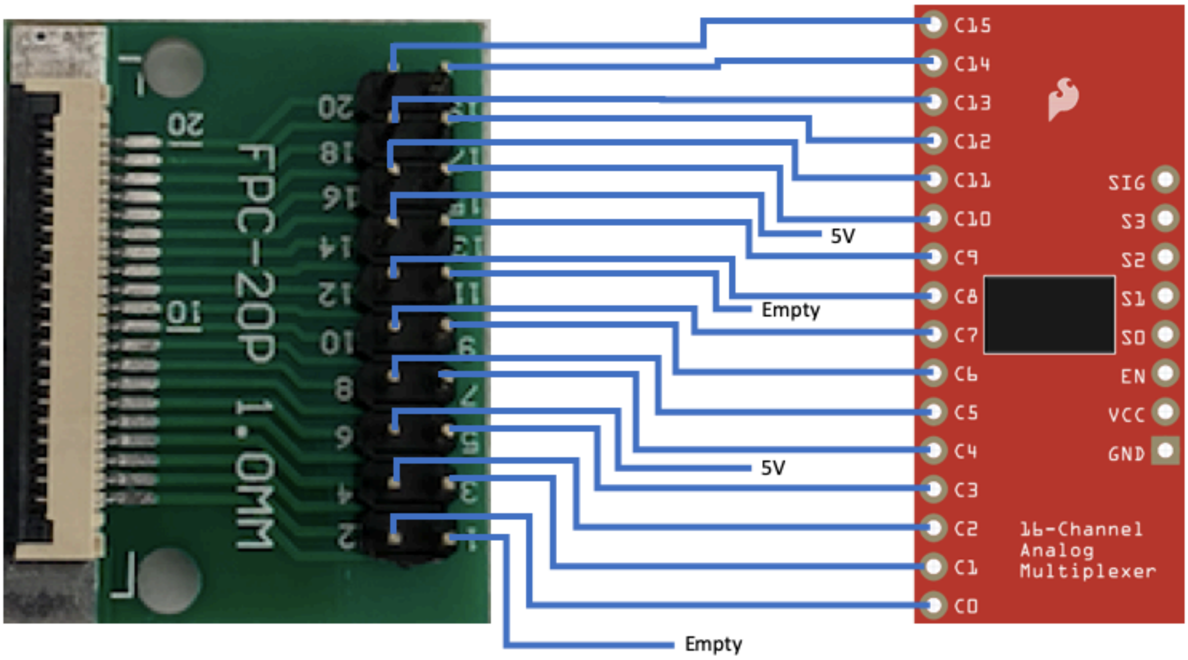


The channels S0, S1, S2 and S3 of the multiplexers are parallel connected to the Arduino channels 8, 9, 10 and 11. The channel Signal from the multiplexer is connected to the Analog channel A0 respectively A1. The analog signals A0 and A1 are also grouted between the analog signal and the ground with a 10kΩ resistor. Both VCC channels are connected to the 5V channel of the Arduino and both GND channels are connected to the GND.
RXD channel of the Bluetooth module is connected to the TXD channel from the Arduino and the TXD channel is connected to the RXD. The VCC signal is connected to the 5V channel and the GND channel is connected to the GND.

The foot pressure sensor:



Connect the foot pressure sensor with the flat cable by using an FCP connector with 20 pins (1.00 mm).
Channel 1 and 11 is empty.
Channel 6 and 14 for the power supply (5V) and the other channels are for the analog signals and must connected to the multiplexer.



Pin FCP Adapter	Pin Multiplexer
1	Empty
2	0
3	1
4	2
5	3
6	5V (VCC)
7	4
8	5
9	6
10	7
11	Empty
12	8
13	9
14	5V (VCC)
15	10
16	11
17	12
18	13
19	14
20	15

The Arduino program:

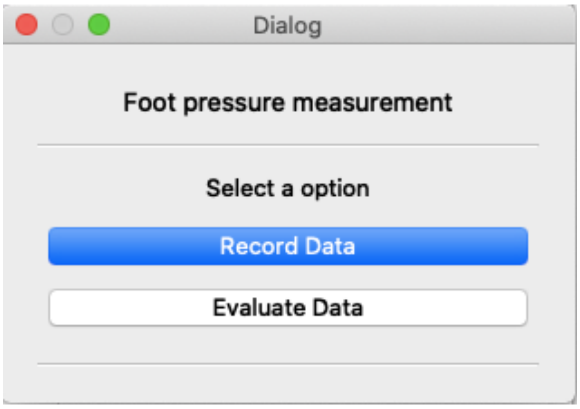
You can copy the code to the Arduino program. For using the Bluetooth module, you don´t need additional changes of the program. You can use the program with USB or Bluetooth connection. For using the Bluetooth module, here is a link with a description for mac: https://alexbloggt.com/arduino-bluetooth/

```
int pin_Out_S0 = 8;
int pin_Out_S1 = 9;
int pin_Out_S2 = 10;
int pin_Out_S3 = 11;
int pin_In_Mux1 = A0;
int Mux1_State[17] = {0};

int pin_In_Mux2 = A1;
int Mux2_State[17] = {0};


void setup() {
  pinMode(pin_Out_S0, OUTPUT);
  pinMode(pin_Out_S1, OUTPUT);
  pinMode(pin_Out_S2, OUTPUT);
  pinMode(pin_Out_S3, OUTPUT);

  //pinMode(pin_In_Mux1, INPUT);
  Serial.begin(9600);
}

void loop() {
  updateMux();
  for(int i = 0; i < 17; i ++) {
    if(i == 16) {
      int pin_In_Mux1 = A0;
      int pin_In_Mux2 = A1;
      Serial.println();


    } else {

      //Serial.print(Mux1_State[i]);
      Serial.print((Mux1_State[i])  * 0.5 , 0);
      Serial.print(";");
      Serial.print(round(Mux2_State[i]) * 0.9, 0);
      Serial.print(";");
    }
  }
}

void updateMux () {
  for (int i = 0; i < 16; i++){
    digitalWrite(pin_Out_S0, HIGH && (i & B00000001));
    digitalWrite(pin_Out_S1, HIGH && (i & B00000010));
    digitalWrite(pin_Out_S2, HIGH && (i & B00000100));
    digitalWrite(pin_Out_S3, HIGH && (i & B00001000));

    Mux1_State[i] = analogRead(pin_In_Mux1);
    Mux2_State[i] = analogRead(pin_In_Mux2);
  }
}
```

User Interface

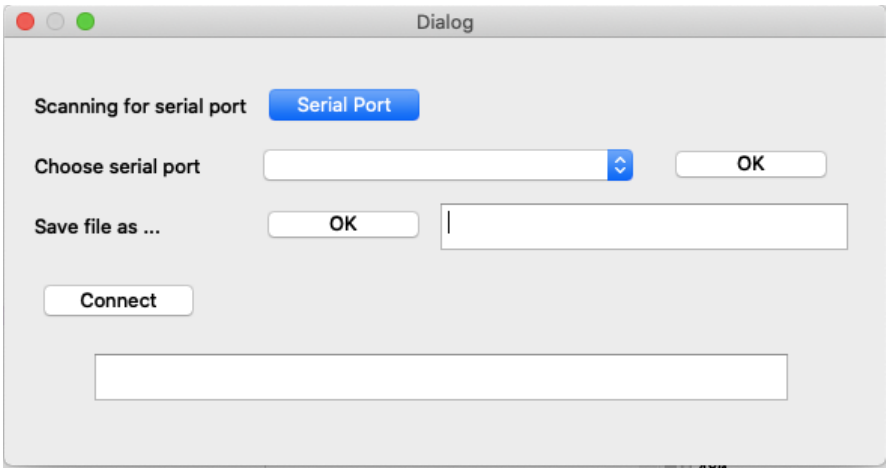
The UI is designed in QT designer.

When you open the program. The program will ask you if you want to record data or do you want to evaluate data.



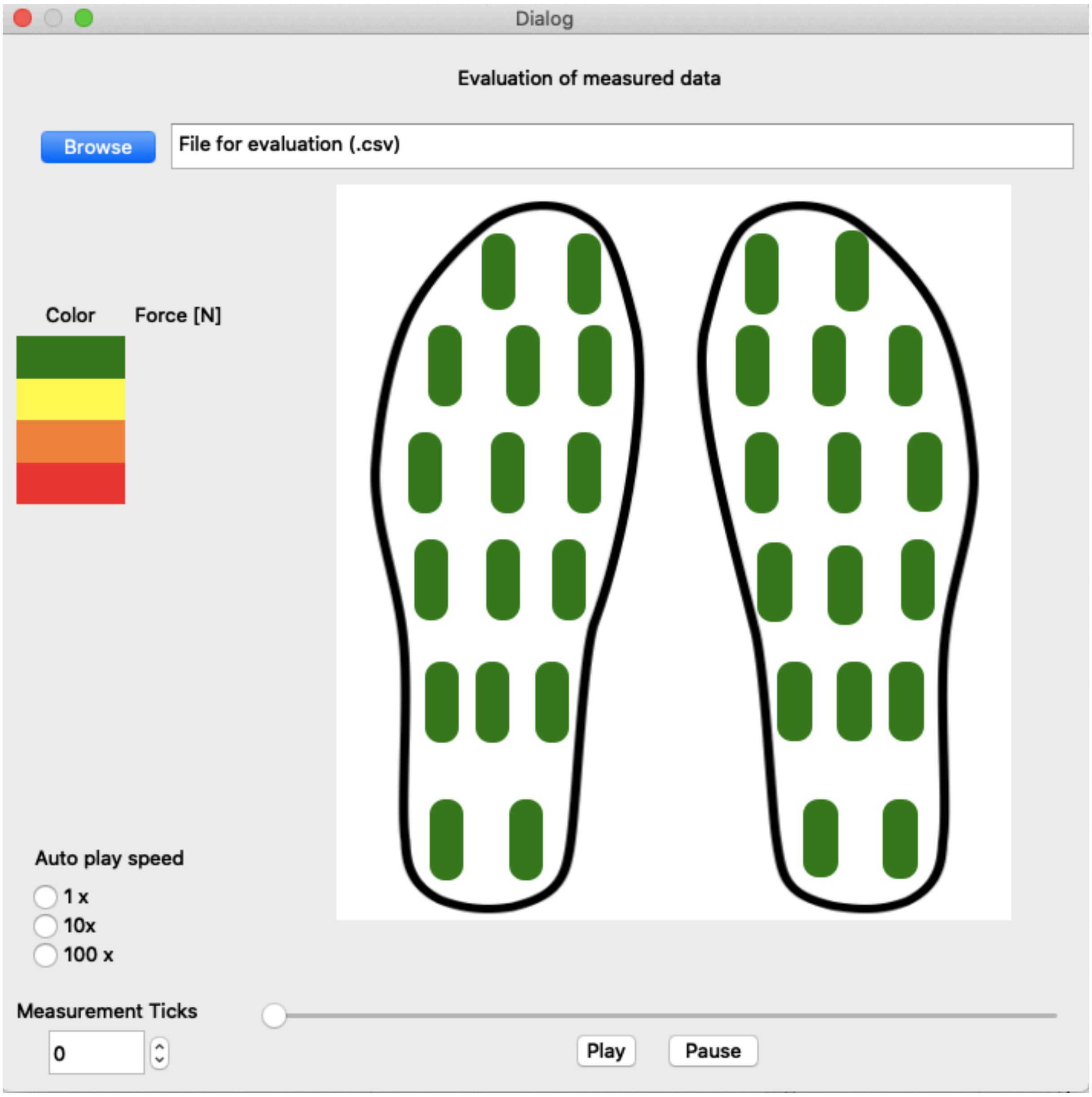
Record Data

If you select record data, this window will be open:
First: You have to click on the serial port button
Second: In the drop down menu select the port where the Arduino is connected to the computer and press OK
Third: It will be creating a CSV file you have to choose the folder and the name of the CSV file
Forth: Click connect and the computer will connect to the Arduino and start the record. After you click connect the button name change to disconnect. Click the button again and stop the measurement.



Evaluation of the measurement data:

First: Click browse and select the CSV file
The filed with the forces will filled automatically. Before you push the play button you have to select the auto play speed. You can also use the slider. The green buttons will change the color and you can see your recorded foot pressure.



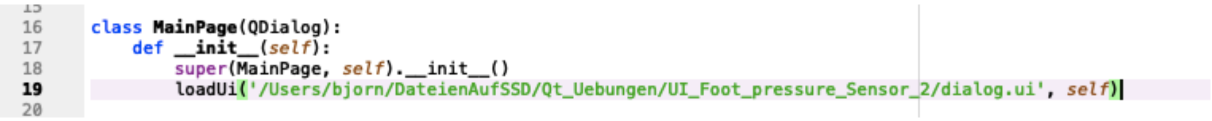
The Python programs

It consists 3 programs:

1. Main Window
2. Dialog Evaluate
3. Dialog Record

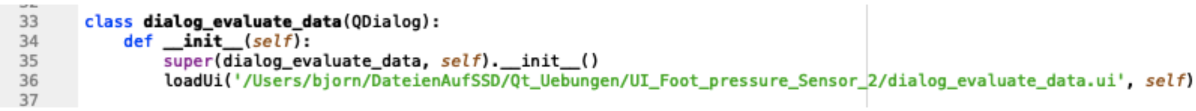
The Qt file and the python file have to be in the same folder.

Main Window:



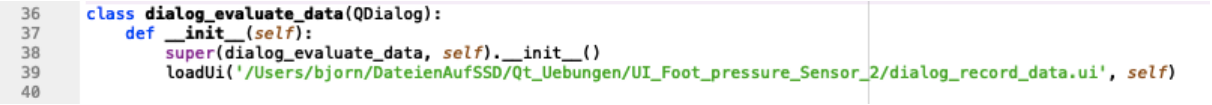
```
class MainPage(QDialog):
    def __init__(self):
        super(MainPage, self).__init__()
        loadUi('/Users/bjorn/DateienAufSSD/Qt_Uebungen/UI_Foot_pressure_Sensor_2/dialog.ui', self)
```

Dialog Evaluate:



```
class dialog_evaluate_data(QDialog):
    def __init__(self):
        super(dialog_evaluate_data, self).__init__()
        loadUi('/Users/bjorn/DateienAufSSD/Qt_Uebungen/UI_Foot_pressure_Sensor_2/dialog_evaluate_data.ui', self)
```

Dialog Record



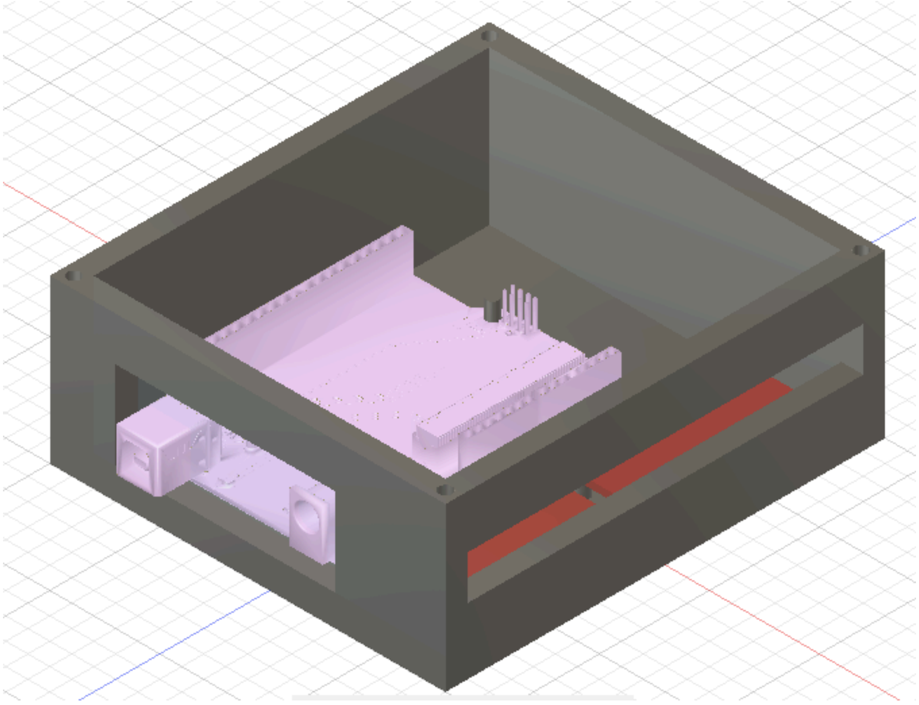
```
class dialog_evaluate_data(QDialog):
    def __init__(self):
        super(dialog_evaluate_data, self).__init__()
        loadUi('/Users/bjorn/DateienAufSSD/Qt_Uebungen/UI_Foot_pressure_Sensor_2/dialog_record_data.ui', self)
```

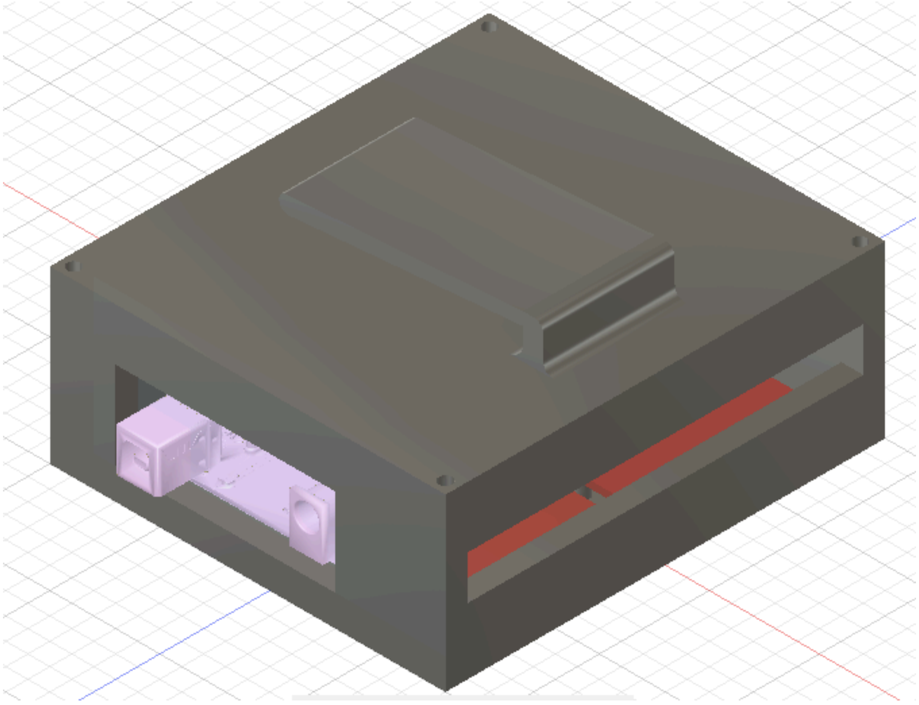
In the field “loadUI” you have to change the folder where the files are located.

Cases

You can download the files for 3d printing by Thingiverse
Link: https://www.thingiverse.com/thing:4694794/apps

Arduino, Bluetooth and Multiplexer case





FCP connector case: