

# LYD-Kit - Modul 1 efterår 2019

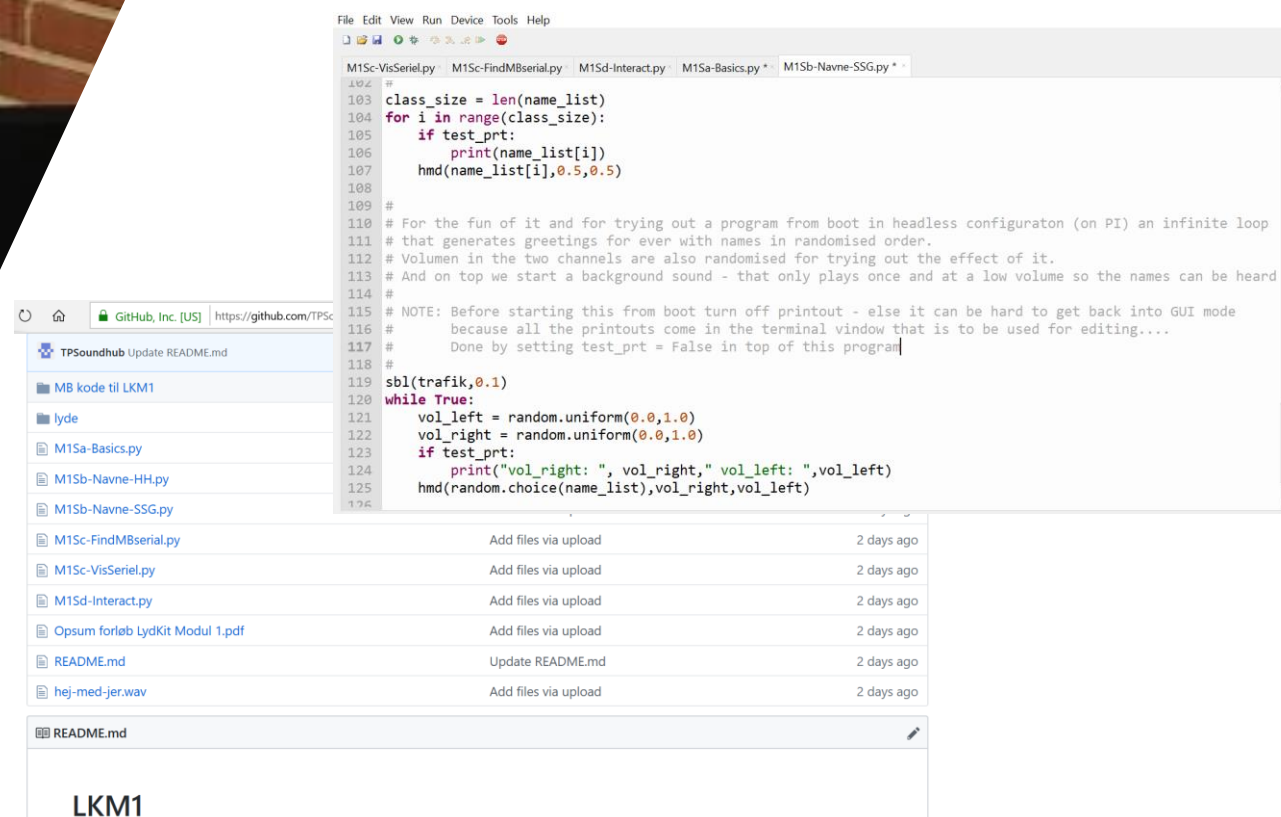
Projekt støttet af:

**midt**  
regionmidtjylland

Integreret udviklingsmiljø (IDE) - Thonny  
Python og biblioteker

- Pygame, Time, Serial, Random

Micro:Bit med kode til input events  
PI , Beocreate boards samt et par højtalere



```
File Edit View Run Device Tools Help
M1Sc-VisSerial.py M1Sc-FindMBserial.py M1Sd-Interact.py M1Sa-Basics.py * M1Sb-Navne-SSG.py *
103 class_size = len(name_list)
104 for i in range(class_size):
105     if test_prt:
106         print(name_list[i])
107         hmd(name_list[i],0.5,0.5)
108
109 #
110 # For the fun of it and for trying out a program from boot in headless configuraton (on PI) an infinite loop
111 # that generates greetings for ever with names in randomised order.
112 # Volumen in the two channels are also randomised for trying out the effect of it.
113 # And on top we start a background sound - that only plays once and at a low volume so the names can be heard
114 #
115 # NOTE: Before starting this from boot turn off printout - else it can be hard to get back into GUI mode
116 # because all the printouts come in the terminal window that is to be used for editing...
117 # Done by setting test_prt = False in top of this program
118
119 sbl(trafik,0.1)
120 while True:
121     vol_left = random.uniform(0.0,1.0)
122     vol_right = random.uniform(0.0,1.0)
123     if test_prt:
124         print("vol_right: ", vol_right," vol_left: ",vol_left)
125     hmd(random.choice(name_list),vol_right,vol_left)
```

TPSoundhub Update README.md

MB kode til LKM1

lyde

- M1Sa-Basics.py
- M1Sb-Navne-HH.py
- M1Sb-Navne-SSG.py
- M1Sc-FindMBserial.py
- M1Sc-VisSerial.py
- M1Sd-Interact.py
- Opsum forløb LydKit Modul 1.pdf
- README.md
- hej-med-jer.wav

Add files via upload	2 days ago
Add files via upload	2 days ago
Add files via upload	2 days ago
Add files via upload	2 days ago
Update README.md	2 days ago
Add files via upload	2 days ago

README.md

LKM1

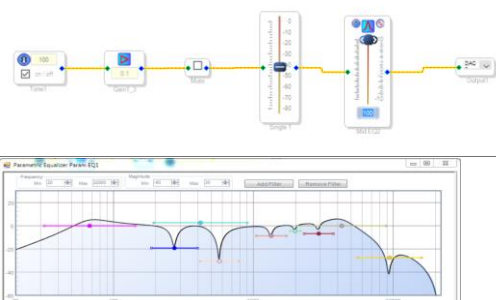
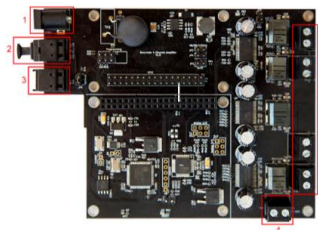
# Materiallesæt

P1 – efteråret 2019

Projekt støttet af:

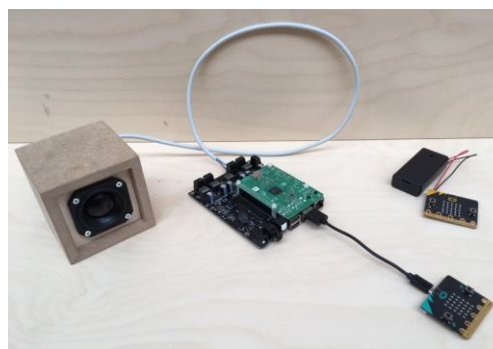
**midt**  
regionmidtjylland

Forstærker modul med DSP



Visuel programmering af DSP

Højttaler og Micro:Bit



Raspberry PI med general purpose uP



```
File Edit Format Run Options Window Help
# Import af biblioteker med ekstra funktioner
# Afspil af "Hello World" som lyd

import pygame
import time

Helloworld = "helloworld.wav"

# initialisering af mixer
pygame.mixer.init()
# spil lyden med bestemt volumen
pygame.mixer.music.set_volume(0.1)
pygame.mixer.music.load(Helloworld)
pygame.mixer.music.play(0)

# Vent til lyden er afspillet.
time.sleep(5)

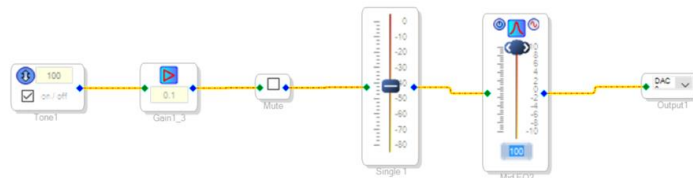
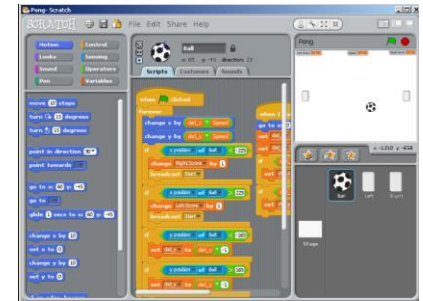
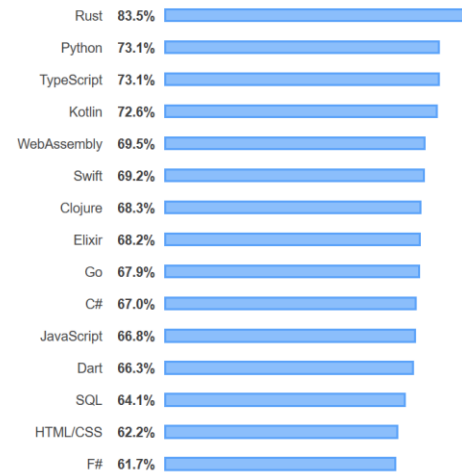
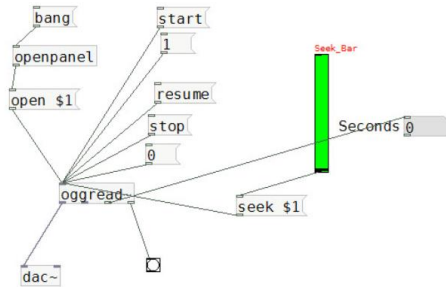
# Ryd op i mixer
pygame.quit()
```

Tekst baseret programmering med lyd

# Programmeringssprog

Projekt støttet af:

**midt**  
regionmidtjylland



For the fourth year in a row, Rust is the most loved programming language among our respondents, followed close behind by Python, the fastest-growing major language today. This means that proportionally, more developers want to continue working with these than other languages.

VBA and Objective-C rank as the most dreaded languages this year. Most dreaded means that a high percentage of developers who are currently using these technologies express no interest in continuing to do so.

Python is the most wanted language for the third year in a row, meaning that developers who do not yet use it say they want to learn it.

text file named HelloWorld.java

```

public class HelloWorld {
    public static void main(String[] args) {
        // Prints "Hello, World" in the terminal window.
        System.out.print("Hello, World");
    }
}

```

name: HelloWorld  
main() method: main  
statements: System.out.print("Hello, World");  
body: { ... }

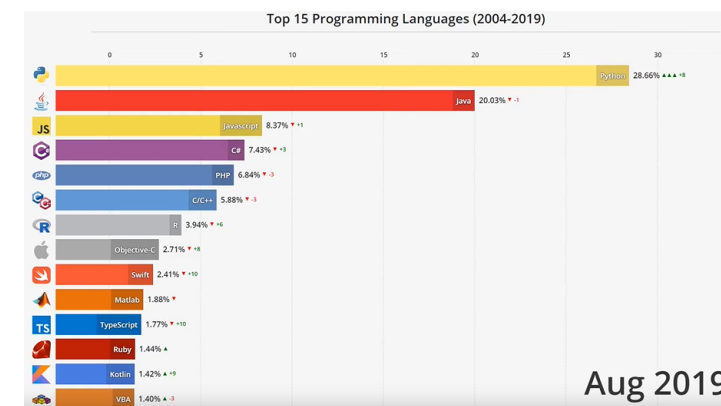
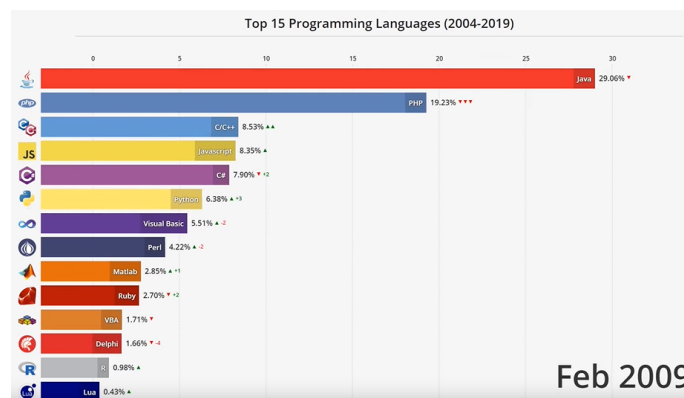
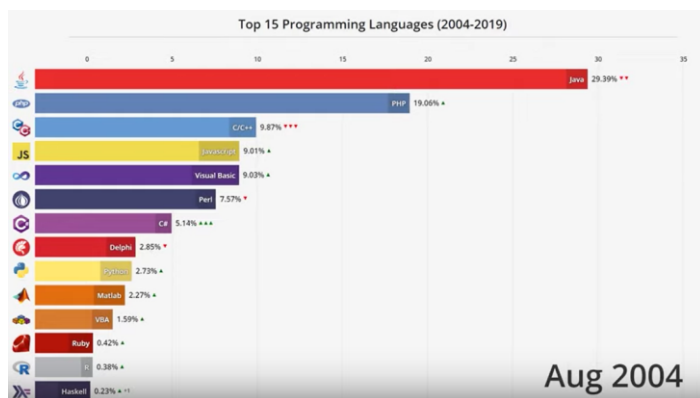
<https://insights.stackoverflow.com/survey/2019#most-loved-dreaded-and-wanted>

# Programmeringssprog

Projekt støttet af:



Statistikker og lignende optællinger har altid nogle forudsætninger så man kan finde forskellige bud på ex. hvad der er mest udbredt så vær varsom. Men de 2 der er i denne præsentation indikerer at Python er udbredt og er blevet mere og mere anvedet over tid.



<https://www.youtube.com/watch?v=yL704C1PI4o>

# LKM1: Hands-on

<https://thonny.org/>

<https://github.com/TPSoundhub/LKM1>

Mere om Python kan I finde her:

<https://docs.python.org/3/>

Mere om Pygame biblioteket finder I her: (fokus på MUSIC og MIXER i denne omgang)

<https://www.pygame.org/docs/ref/music.html>

Generelt så kan I få meget ud af at stille 'Google' spørgsmål på engelsk. Ex.:

"How to make loop in python".

"How to generate random number in python"

Etc....

Så kommer der ofte gode eksempler i diverse links. Kan dog anbefale et par steder til introduktion:

<https://www.w3schools.com/python/default.asp>

<https://www.learnpython.org/en/>

<https://wiki.python.org/moin/>

<https://realpython.com/>

# LKM1: Første program

Kører og ser at der kommer "hej med jer" ud i shell

Første lille funktion som så kan kaldes flere gange og gøre koden 'effektiv' Kaldes med pn("xxx") flere gange

En anden funktion med tal i stedet for tekst.  
Introduktion til betingelse if/elif/else

":" og indrykning vigtige for at 'python'/Thonny kan forstå det!! Det som er indrykket hører til funktion/betingelse

```
print ("hej med jer")

def pn(t):
    print("Hej med dig: ")
    print(t)
    print("Godt at se dig")
    print("")

def lms(n):
    if n<10:
        print("tal under 10")
    elif n == 10:
        print("tal er 10")
    else:
        print("tal over 10")
```



# LKM1: Introduktion til lister/løkker

Køre og se at man kan pille et tegn ud af en streng

At man kan lave en liste med strenge

Og se at man kan udpege en enkelt i listen

Fra 0 til 3 når der er 4 i listen.

Bruger funktionen "pn" som blev defineret i forrige trin

At man kan lave en for-løkke som går igennem alle elementerne i en liste og udføre en funktion derpå

Kan pille flere end et tegn ud af en streng ved at udpege et område  
Prøv ex. `print(jens[1:3])`

```
jens = "Jens"
print(jens[2])

nl = ["Jens", "Peter", "Lotte", "Marie"]

print(nl[3])

pn(nl[1])

for i in range(4):      # da vi ved listen har 4 indgange.
    pn(nl[i])           # Kunne også bruge len(nl) som er mere generel.
```

# LKM1: Introduktion til biblioteker

Der findes masser af foruddefinerede funktioner i biblioteker man kan importere og bruge i sin egen kode.  
Her time og random til at lave stop/pauser i program samt til at udvælge/generere tilfældigt

```
import random  
import time
```

En while-løkke som fortsætter så længe betingelsen er sand.  
Her sat til True så den aldrig stopper!  
Bruger random.choice til at vælge en tilfældig indgang i tabel fra forrige trin.  
og funktion pn fra første trin til at udskrive.  
Bruger time.sleep(5) til at holde en pause på 5 sekunder mellem hvert kald af funktionen pn

```
while True:  
    pn(random.choice(nl))  
    time.sleep(5)
```



# LKM1: Introduktion til pygame(sound)

```
import pygame

pygame.mixer.init()
pygame.mixer.music.set_volume(0.5)
pygame.mixer.music.load("hej-med-jer.wav") # works with the file in same directory as the program else use full path name
pygame.mixer.music.play()
```

Bruger mixer i pygame biblioteket til at starte afspilning af en længere lyd-fil med halv volume.  
Kan ex. bruges til at starte en baggrunds lyd.

Pygame.mixer.init() skal bare kaldes én gang i begyndelsen af programmet.  
Resten kan bruges flere gange til at starte flere lyde. Man kan evt. lave en funktion til det

## BEMÆRK:

At når lydfilen er i samme direktorier som det program man skriver, så er der ikke behov for sti-navnet. Ligger lyd filen et andet sted skal man have udpeget lydfilen med det fuld sti navn.

Så er der problemer med at finde den rigtige sti så kan man 'bare' gemme lyd filer samme sted som programmerne.

# LKM1: Introduktion til pygame(sound)

Definerer en funktion der bruger mixer i pygame til at :

- Finde en kanal (i stereo)
  - Sætte volume i hhv. højre og venstre udgang for kanal
  - Spiller én lyd udpeget med "hejmeddig" som lydobjekt (der kan være flere samtidigt og 'oveni' det som spiller i 'music' fra forrige trin)
  - Venter 1 sek til lyd er færdig  
(Da vi ved at det er en kort lyd 😊. Kan laves smartere ved at spørge til længden af lyden)
  - Gentager samme historie med en lyd der kommer til funktionen som parameter.
- 
- Programmet **M1Sb-Navne-SSG.py** (i Struer) eller **M1Sb-Navne-HH.py** (i Herning) kombinerer tingene til at lave navneopråb for klassen først i alfabetisk rækkefølge, dernæst i en uendelig løkke med tilfældig rækkefølge og tilfældig volume i hhv. højre og venstre højttaler. (programmet ligger i jeres LKM1 folder downloadet fra Github.

```
import pygame
```

```
def hmd(n,vl,vr):  
    channel = pygame.mixer.find_channel()  
    channel.set_volume(vl,vr)  
    channel.play(pygame.mixer.Sound(hejmeddig))  
    time.sleep(1)  
    channel = pygame.mixer.find_channel()  
    channel.set_volume(vr,vl)  
    channel.play(pygame.mixer.Sound(n))  
    time.sleep(1)
```

# LKM1: Introduktion Micro:Bit/Serial

Åbn den serielle kanal som MB er på!  
Find den med **M1Sc-VisSeriel.py**

Funktion til at læse tegn fra den serielle.  
Med return afleverer funktionen tegn modtaget fra Micro:Bit  
Der modtages en hel tekst streng som også indeholder navn.

Uendelig løkke der læser tegn fra Micro:Bit og udskriver den

```
import serial

ser = serial.Serial()

ser.baudrate = 115200
ser.port = "COM6" # på MAC: "/dev/tty.usbmodem14102"
                # eller   "/dev/cu.usbmodem14102"

ser.open()
```

```
test_print = True

def receive_char():
    microbitdata = str(ser.readline())
    if test_print:
        print(microbitdata)
    return microbitdata[2]
```

```
while True:
    rc = receive_char()
    if test_print:
        print("Modtaget tegn: ",rc)
    ...
```

# LKM1: Input's fra Micro:Bit

```
# Microbit functionality in short:
#
# Send string "Hello world ..." on serial USB and make Happy face on LEDs when powerup
# Send characters on serial USB when sensors/buttons are triggered
#
# - "A", "B" when a/b key is pressed - plus "A" "B" on local display - Once pr press
# - "C" when pin 0 is activated - repetitive as long as pin 0 is active with 500ms spacing
# - "O" - "G" randomly selected when pin 1 is active - repetitive 500ms spacing
# - "V" - when tilted to left - repetitive 500ms spacing
# - "H" - When tilted to right - repetitive 500ms spacing
# - "F" - when tilted forward - repetitive 500ms spacing
# - "T" - when tilted backwards - repetitive 500ms spacing
# - "D" - when light intensity goes from outside range into range between 40 and 180. (Skift til Dag)
# - "N" - when light intensity goes from above to below 40. (Skift til Nat)
# - "S" - when light intensity goes from below to above 180. (Skift til Solskin>Dag)
#
# - Switch mode between sending on radio vs sending on serial when pin 2 is active
# - small square followed by big square -> Radio transmission from remote location
# - big square followed by small square -> Serial transmission local connected via USB
# If transmission on radio then feedback on LED are in small letters "a","b" etc..
# - and so is the character transmitted.
# - Digits not transmitted - so no action on pin 1 when in radio transmission mode.
#
# If anything is received on radio the received character is sent on serial USB. No feedback on LEDs
# Feedback is given on the remote Micro:Bit
#
# All characters are followed by name the one having the Micro:Bit
```

# LKM1: BYG

Kombiner og byg -

Lav kode der afspiller lyde afhængigt af input fra Micro:Bitten

Definer lydene – konstanter med reference til lydfileerne inklusiv fuld sti med mindre de ligger samme sted som programmet I laver.

Brug funktionerne `receive_char` og `play_sound` til at lave noget som reagerer på input fra Micro:Bitten i en uendelig løkke der læser tegn fra `micro:bit` og afhængig af det modtagne tegn spiller en lyd. Brug evt. programmet **M1Sd-Interact.py** som skabelon (ligger i folderen LKM1 downloadet fra Github)

```
def play_sound(n,vl,vr):
    channel = pygame.mixer.find_channel()
    channel.set_volume(vl,vr)
    channel.play(pygame.mixer.Sound(n))

def receive_char():
    microbitdata =str(ser.readline())
    if test_print:
        print(microbitdata)
    return microbitdata[2]
```

```
pygame.mixer.init()
music_volume = 0.1
```

```
pygame.mixer.music.load(baggrundslyd)
pygame.mixer.music.set_volume(music_volume)
pygame.mixer.music.play(-1)
```

```
while True:
    rc = receive_char()
    print(rc)
    if rc == "A":
        play_sound(lyd1,0.1,0.7)
    elif rc == "B":
        ....
```

. Byg videre I kan evt få fat i navnet på den der sender i `receive_char` (hvordan tager man en del streng ud af en længere streng?)

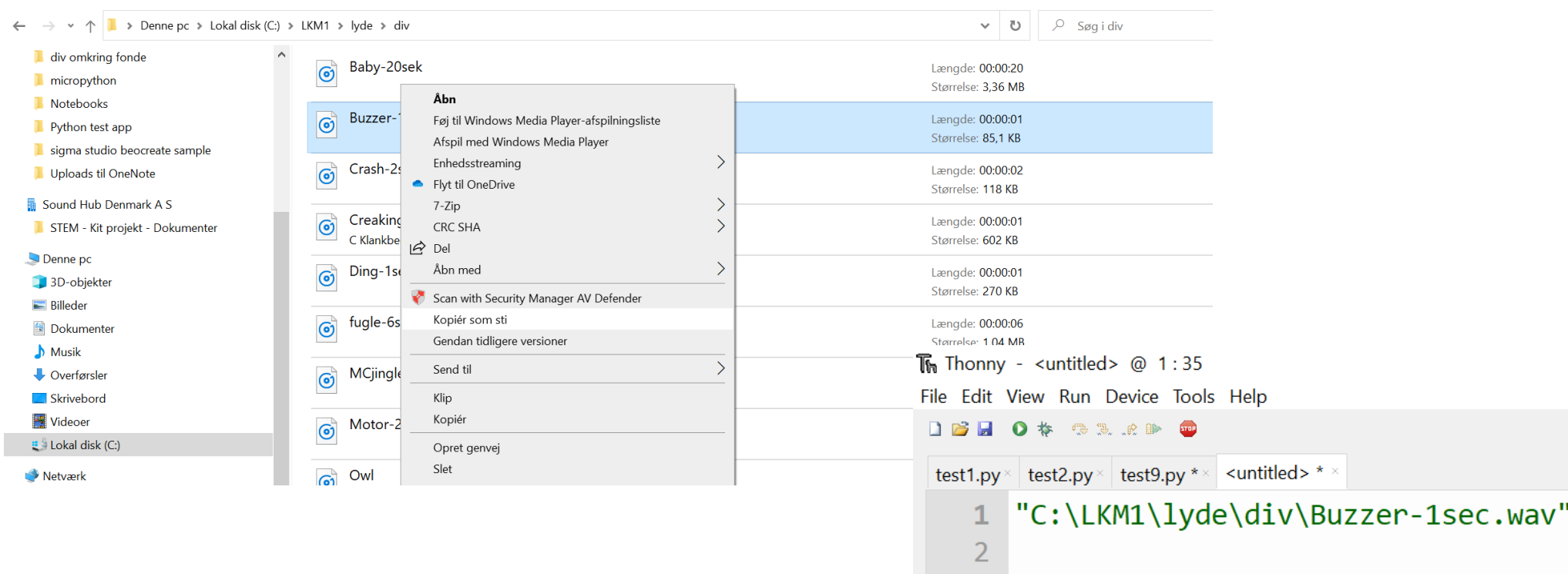
# LKM1: FAQ – Sti på PC (windows 10)

Hvordan finder jeg den rigtige sti på PC

Find placering/fil i file explorer. Marker filen. Tryk og hold shift samtidigt med højreklik. Så kommer kontekst menu med mulighed for at kopiere sti. Brug ctrl-v i Thonny til at få fuld sti incl. fil navn kopieret ind.

**BEMÆRK at sti så er med backslash** – Det bør ændres til forwardslash eller dobbeltslash for ikke at få andre problemer!!

Se selvstændig slide om det.



The screenshot shows a Windows File Explorer window with the address bar set to 'Lokal disk (C:) > LKM1 > lyde > div'. The left sidebar shows the 'Lokal disk (C:)' selected. The main pane displays a list of audio files. The file 'Buzzer-1sec.wav' is selected, and a context menu is open over it. The menu options include 'Åbn', 'Føj til Windows Media Player-afspilningsliste', 'Afspil med Windows Media Player', 'Enhedsstreaming', 'Flyt til OneDrive', '7-Zip', 'CRC SHA', 'Del', 'Åbn med', 'Scan with Security Manager AV Defender', 'Kopiér som sti', 'Gendan tidligere versioner', 'Send til', 'Klip', 'Kopiér', 'Opret genvej', and 'Slet'. The 'Kopiér' option is highlighted. In the bottom right corner, the Thonny IDE is open, showing the menu 'File Edit View Run Device Tools Help'. The file explorer window is titled 'Thonny - <untitled> @ 1:35'. The file explorer window shows the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

In the Thonny IDE, the file explorer window is titled 'Thonny - <untitled> @ 1:35'. The file explorer window shows the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the following files and their details:

File Name	Length	Size
Baby-20sek	00:00:20	3,36 MB
Buzzer-1sec	00:00:01	85,1 KB
Crash-2s	00:00:02	118 KB
Creaking	00:00:01	602 KB
Ding-1s	00:00:01	270 KB
fugle-6s	00:00:06	1,04 MB
MCjingle		
Motor-2		
Owl		

The Thonny IDE shows the file explorer window with the

# LKM1: FAQ – Sti på MAC (OSX)

Hvordan finder jeg den rigtige sti på MAC

Find fil og marker den – Brug context menu (to fingre og tryk ??). Brug kopi som sti. Kopier til Thonny.

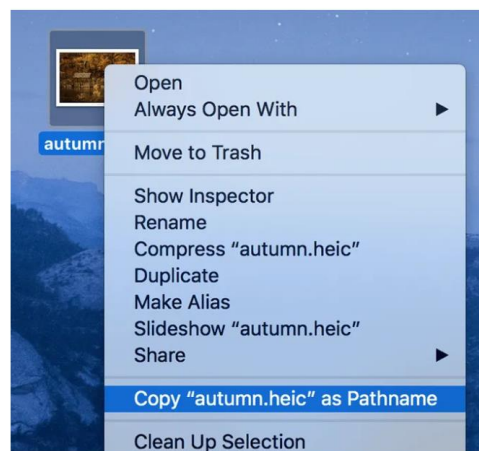
**CHECK om sti så er med backslash** – Det bør ændres til forwardslash eller dobbeltslash for ikke at få andre problemer!!

Se selvstændig slide om det.

## Copy File Paths From The Context Menu

The context menu on your Mac is a really powerful tool as [it lets you do much more with your files than just letting you rename or delete your files](#).

One of the useful and hidden options in your context menu lets you directly copy the path of a file to your clipboard.



Since it's hidden by default, it won't appear when you right-click on a file on your Mac. However, unhiding the option is pretty easy and all it takes is pressing and holding down the Option key. This will make the option visible in your context menu.

To use the option, right-click on a file in the Finder, hold down the **Option** key on your keyboard, and you'll see the **Copy "file-name.ext" as Pathname** option. Click on it to copy your file's path.

It will have copied the path of your selected file as plain text to your clipboard.



# LKM1: FAQ – forward “/” eller backslash “\”

Er der forskel på at bruge forward- eller backward slash i sti navne når lydfiler skal udpeges?

JA- Backslash udpeger specialtegn i kombination med andre tegn så den skal man være forsigtig med  
– Kør M1Sb-slashtest.py i Thonny fra LKM1-master direktoriet (hentet fra Github i seneste version) til illustration.

Escape Sequence	Meaning
<code>\newline</code>	Ignored
<code>\\</code>	Backslash (\)
<code>\'</code>	Single quote (')
<code>\"</code>	Double quote (")
<code>\a</code>	ASCII Bell (BEL)
<code>\b</code>	ASCII Backspace (BS)
<code>\f</code>	ASCII Formfeed (FF)
<code>\n</code>	ASCII Linefeed (LF)
<code>\r</code>	ASCII Carriage Return (CR)
<code>\t</code>	ASCII Horizontal Tab (TAB)
<code>\v</code>	ASCII Vertical Tab (VT)
<code>\ooo</code>	ASCII character with octal value <i>ooo</i>
<code>\xhh...</code>	ASCII character with hex value <i>hh...</i>

```
#
# Kode stump til at illustrere forskel med backslash "\ " og forwardslash "/"
#
# Det ser umiddelbart meget tilforladeligt ud. MEN bemærk udskriften for t6
#   - Ikke som forventet backslash forsvinder!!!
#   - Det er fordi \b tolkes som et special tegn - som backspace BS!!!!
#
# Og bemærk også udskrift for t13-t18
#   - Der kommer kun een backslash i alle udskrifter selvom der står to efter hinanden i konstanterne!
#
# Bemærk som det sidste at det kan ikke lade sig gøre at slutte med backslash
#   - så får man en fejl - prøv det!!
#
#
# Konklusion - Brug forward slash eller dobbelt backslash i sti navne!!
#
# Hvorfor kommer man til at bruge backslash?
#   - Fordi det er det man får når man kopierer sti navn i fil/folder system!
#   - Hvorfor virker det nogle gange med enkelt backslash?
#   - Fordi det afhænger af det tegn/bogstav som kommer efter!
#   - se hvilke i https://docs.python.org/2.0/ref/strings.html
#
```

```
t1 = "tekst streng med backslash \ efterfulgt af space"
t2 = "tekst streng med backslash \L efterfulgt af L som i LKM1"
t3 = "tekst streng med backslash \l efterfulgt af l som i lyde"
t4 = "tekst streng med backslash \d efterfulgt af d som i div"
t5 = "tekst streng med backslash \h efterfulgt af h som i hej-ol"
t6 = "tekst streng med backslash \b efterfulgt af b som i baggrund"

t7 = "tekst streng med forwardslash / efterfulgt af space"
t8 = "tekst streng med forwardslash /L efterfulgt af L som i LKM1"
t9 = "tekst streng med forwardslash /l efterfulgt af l som i lyde"
t10 = "tekst streng med forwardslash /d efterfulgt af d som i div"
t11 = "tekst streng med forwardslash /h efterfulgt af h som i hej-ol"
t12 = "tekst streng med forwardslash /b efterfulgt af b som i baggrund"

t13 = "tekst streng med dobbelt backslash \\ efterfulgt af space"
t14 = "tekst streng med dobbelt backslash \\L efterfulgt af L som i LKM1"
t15 = "tekst streng med dobbelt backslash \\l efterfulgt af l som i lyde"
t16 = "tekst streng med dobbelt backslash \\d efterfulgt af d som i div"
t17 = "tekst streng med dobbelt backslash \\h efterfulgt af h som i hej-ol"
t18 = "tekst streng med dobbelt backslash \\b efterfulgt af b som i baggrund"

nl = [t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12,t13,t14,t15,t16,t17,t18]
```

```
for i in range(len(nl)):
    print("t",i+1," : ",nl[i])
```

Python 3.7.5 (bundled)

```
>>> %Run slashtest.py
```

```
t 1 : tekst streng med backslash \ efterfulgt af space
t 2 : tekst streng med backslash \L efterfulgt af L som i LKM1
t 3 : tekst streng med backslash \l efterfulgt af l som i lyde
t 4 : tekst streng med backslash \d efterfulgt af d som i div
t 5 : tekst streng med backslash \h efterfulgt af h som i hej-ol
t 6 : tekst streng med backslash efterfulgt af b som i baggrund
t 7 : tekst streng med forwardslash / efterfulgt af space
t 8 : tekst streng med forwardslash /L efterfulgt af L som i LKM1
t 9 : tekst streng med forwardslash /l efterfulgt af l som i lyde
t 10 : tekst streng med forwardslash /d efterfulgt af d som i div
t 11 : tekst streng med forwardslash /h efterfulgt af h som i hej-ol
t 12 : tekst streng med forwardslash /b efterfulgt af b som i baggrund
t 13 : tekst streng med dobbelt backslash \\ efterfulgt af space
t 14 : tekst streng med dobbelt backslash \\L efterfulgt af L som i LKM1
t 15 : tekst streng med dobbelt backslash \\l efterfulgt af l som i lyde
t 16 : tekst streng med dobbelt backslash \\d efterfulgt af d som i div
t 17 : tekst streng med dobbelt backslash \\h efterfulgt af h som i hej-ol
t 18 : tekst streng med dobbelt backslash \\b efterfulgt af b som i baggrund
```

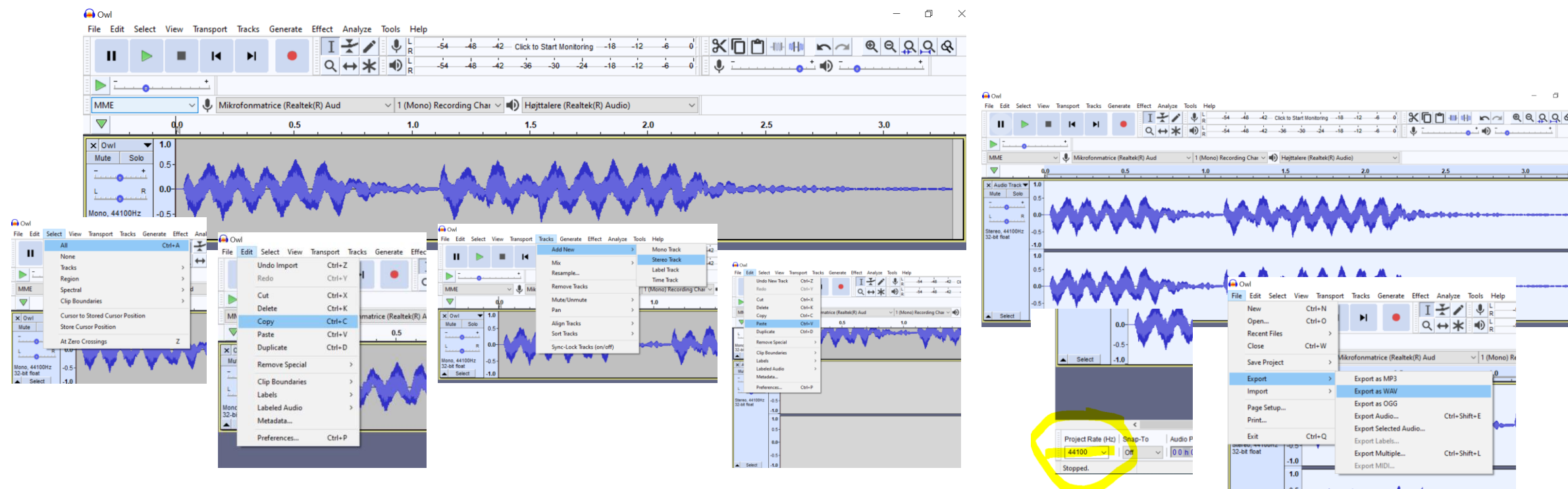
# LKM1: FAQ – Kan man bruge alle lydfiler

NEJ – Pygame mixer music/Sound kan ikke afspille alle lydformater. Brug wav filer!

BEMÆRK at ikke alle wav filer virker, og der kan også være forskel på hvilke der virker på hhv. PC/MAC og PI!

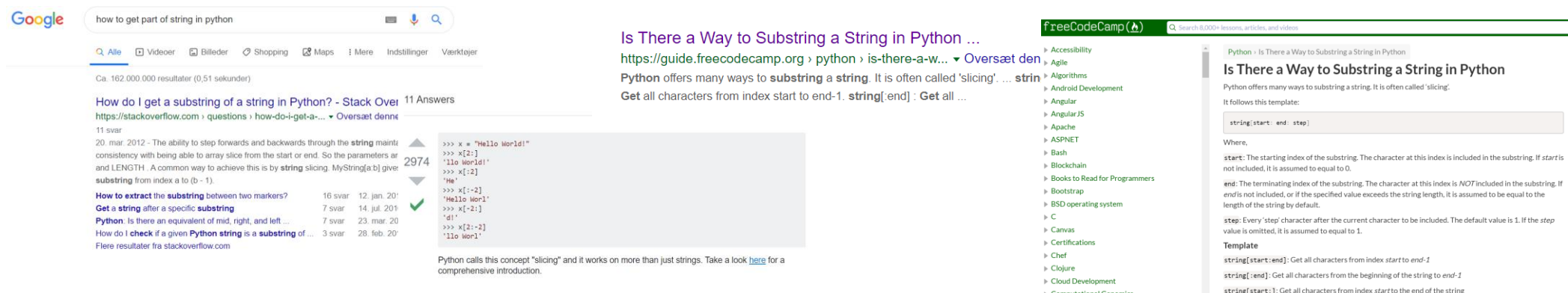
- Ex. Mono fil virker på PC men ikke på PI, Der kan være problemer med forskellige sampling rates etc. Stereo i 44100Hz virker på begge.

PRØV – hvis det ikke virker så brug ex. Audacity til at konvertere så det kommer til at fungere



# LKM1: FAQ – Hvordan finder navn på den der sender

SØG inspiration – Prøv at spørge google med : ” how to get part of string in python”  
check svarene og se om ikke der er noget inspiration at hente



The image shows two screenshots. The left screenshot is a Google search result for "how to get part of string in python". It displays a list of search results, with the top one being a Stack Overflow question titled "How do I get a substring of a string in Python?". The right screenshot is a FreeCodeCamp article titled "Is There a Way to Substring a String in Python". The article explains that Python offers many ways to substring a string, often called 'slicing'. It provides a template: `string[start: end: step]` and explains the meaning of each parameter: `start` is the starting index (included), `end` is the terminating index (not included), and `step` is the step size (default 1). It also includes a table of common slicing patterns.

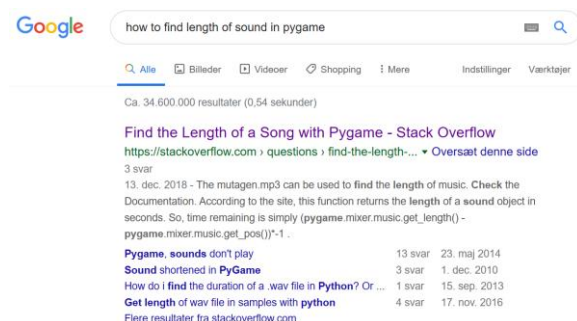
PRØV ting af – prøv med kode nedenfor som udbygning på det der tidligere er lavet:

```
def receive_char():
    microbitdata = str(ser.readline())
    print(microbitdata)
    return microbitdata[2], microbitdata[4:14]

while True:
    rc,hvem = receive_char()
    print("modtaget tegn:",rc,"fra:",hvem, "Streng med længden:",len(hvem))
```

# LKM1: FAQ – Hvordan finder man længden af en lyd

SØG inspiration – Prøv at spørge google med : ” how to find length of sound in pygame”  
check svarene og se om ikke der er noget inspiration at hente



5

You could also load the song as a normal sound and check the length while using mixer.music for playing it.

```
a = pygame.mixer.Sound("test.wav")
print("length", a.get_length())
```

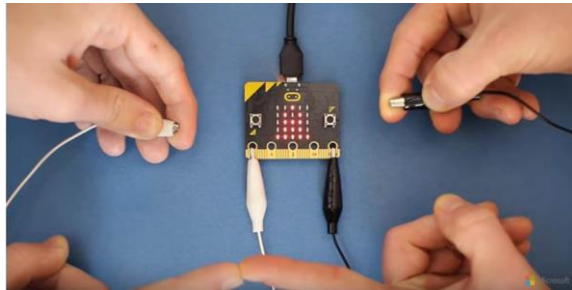
PRØV ting af – prøv med kode nedenfor:

```
def play_sound(n,vr,vl):
    channel=pygame.mixer.find_channel()
    channel.set_volume(vl,vr)
    channel.play(pygame.mixer.Sound(n))
    return pygame.mixer.Sound(n).get_length()

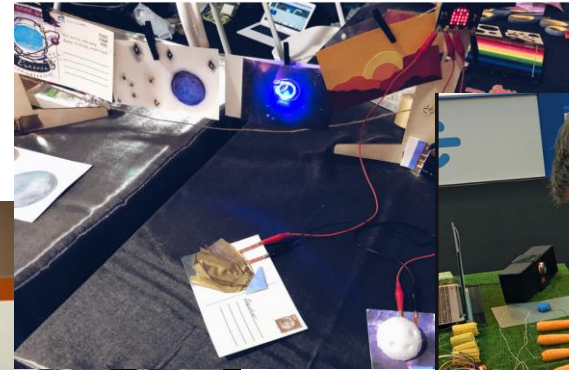
while True:
    rc,hvem = receive_char()
    if rc == "A":
        length_of_sound = play_sound(lyd1,0.5,0.5)
        print("længde af lyd1: ",length_of_sound)
```

# LKM1: FAQ – Kan man selv bygge ‘knapper’

SØG inspiration – JA – det drejer sig om at lave en ‘kortslutning’



<https://youtu.be/GEpZrvbsO7o?t=299>



PRØV ting af – Lav kredsløb der kan kortslutte mellem GND og Pin 0 på Microbitten:

- Brug krokodillenæb og kobbertape til at forbinde mellem det I bygger og pin's på MB
- Brug input event/modtaget tegn "C" eller "c" i jeres kode til at lave en handling på den 'knap' I har bygget.
- Alt hvad der er ledende kan bruges i 'knap' kredsløbet.
- Man kan bygge noget af ikke ledende materiale og så påklister ledende materiale til 'kortslutningen' – ex. kobbertape.
- I kan selvfølgelig også bygge en MB ind i noget fysisk der vipper og bruge "v", "h", "f" og "t". Og ja det skal nok kombineres med at det er en MB som er sat op til at sende på radio og med et batteri. Derfor de 'små' bogstaver som input events.

# LKM1: FAQ – Hvad gør man for at undgå lys input når man trykker på knap

?

"Når man kommer med sin hånd og vil trykke på en knap (A/B) på Microbitten eller man vil tage den op i hånden og vippe med den, så skygger man for lyssensoren og får så genereret en "N" input event. Det er træls og kan genere i vores kode.

Hvad kan vi gøre for at undgå det?

!

Udnyt at der i gruppen er flere MB's og giv hver MB en 'special' opgave. Det kan for eksempel være ved at sige at lys input event fra den MB som bruges til at holde i hånden IKKE skal anvendes i jeres program. Dvs. at I så skal filtrere "S"/"s", "D"/"d" og "N"/"n" input events fra den MB væk i jeres kode. Det gør i ved at teste på navn der kommer fra MB. Altså kommer der et lys input event fra en MB med det bestemte navn så skal der bare ikke ske noget på det input. Se kode eksempel på GitHub – M1Se-Filterinput. Kør det og se hvad det er den kode gør. Den gør flere ting så læs kommentarerne i koden som det første.

!

Man kan også bare blive i den fysiske verden, og tage noget IKKE gennemsigtigt tape og klistre hen over dioderne på Micro:Bitten så vil dioderne ikke 'se' noget lyd og der vil ikke komme nogle lys events fra den Micor bit til jeres kode.

Brug det til at tænke bredere på løsninger. Der er både den fysiske (analoge verden) og den med programmerne inde i maskinen (den digitale) 😊



# LKM1: FAQ – Hvordan ser vi hvilken lysintensitet der er på et bestemt tidspunkt

?

Vi ønsker at lave forskellig lyd/handling på input events som eksempelvis tilt ("V"/"v") afhængig af om lys intensiteten er lav, middel eller høj. Kan vi spørge efter lydintensiteten når vi har brug for den?

!

Nej man kan ikke spørge MB om at returnere lydintensiteten på et vilkårligt tidspunkt. Microbitten sender events når tilstanden skifter mellem lav, mellem og høj på det tidspunkt den registrerer tilstandsskiftet ("N","D","S","n","d","s"). Det I kan gøre er at huske tilstanden lokalt i jeres program når i modtager event om skiftet. I kan gemme det i en global variable og så teste på dens værdi når I har brug for det. Se kode eksempel på GitHub – M1Se-Filterinput. Kør det og se hvad det er den kode gør. Den gør flere ting så læs kommentarerne i koden som det første.

```
ser = serial.Serial(timeout=1)
ser.baudrate = 115200
ser.port = "COM5"
```

```
test_print_1 = False
test_print_2 = True
```

```
light_events = ["N","D","S","n","d","s"]
current_light_state = None
```

```
tilt_events = ["H","V","F","T","h","v","f","t"]
current_tilt_state = None
```

```
relevant_sender_1 = "Knud-3"
relevant_sender_2 = "Knud 1"
```

```
def receive_char():
    microbitdata = str(ser.readline())
    if test_print_1: print(len(microbitdata),microbitdata)
    if len(microbitdata)>3:
        return microbitdata[2], microbitdata[4:14]
    else:
        return None, None

def relevant_event(rc,who):
    if (who == relevant_sender_1) and (rc in light_events):
        return rc
    elif (who == relevant_sender_2) and (rc not in light_events):
        return rc
    else:
        return None
```

```
ser.open()

while True:
    rc,who = receive_char()
    if rc == None: # Nothing received - taht is a timeout on the serial port then
        if test_print_1: print("timeout - nothing received")
        current_tilt_state = None # not tilted as we have a timeout from MB reading.
    else: # Something received
        if test_print_1: print("received character:",rc,"from:",who, "String with the lenght",len(who))
        event = relevant_event(rc,who)
        if event!=None:
            if event in tilt_events:
                if event == current_tilt_state:
                    if test_print_2: print("Do repeated tilt stuff if any",who,event)
                else:
                    tilts_all_since_start +=1
                    if event == "V" or event == "v":
                        tilts_left_since_start += 1
                    if test_print_2:
                        print("Do tilt stuff on first tilt event in specifik direction and set tilt state",who,event)
                        print("The code for tilt can then take current light state into accout if you want", current_light_state)
                        print("No of unique - not repeated - tilts in any direction since programstart:",tilts_all_since_start)
                        print("No of unique - not repeated - Vv tilts since programstart:",tilts_left_since_start)
                        current_tilt_state = event
            elif event in light_events:
                if test_print_2: print("Do light stuff and remember current light state for use elsewhere",who,event)
                current_light_state = event
            else: # Other input than tilt and light
                if event == "A" or "a":
                    Aa_pressed_since_start += 1
                if test_print_2:
                    print("Do stuff on other input than tilt and light",who,event)
                    print("The code for other inputs can then take current light state into accout if you want", current_light_state)
                    print("No of A and a presses since programstart:",Aa_pressed_since_start)
```



# LKM1: FAQ – Hvad gør man hvis man kun har brug for at få at vide MB er vippet

?

Vi vil starte en lyd når Micro:Bitten vippes i én retning, men vi bliver ved med at få input events for vip, så vi kommer til at starte lyden igen og igen, hvilket ikke er det vi ønsker. Vi ønsker kun at lave en handling på det første input event når Micro:Bitten vippes fra normal/vandret position til en bestemt retning. Hvordan kan vi skelne mellem første vip event og efterfølgende?

!

I kan lave kode som husker tilstande på vip retning (tilt) og filtrere input event afhængig af hvordan tilstanden er. Så hvis tilstanden er at den er vippet til venstre så betyder input events "V" og "v" at der er efterfølgende vip events (ikke det første – og de kan så filtreres væk i jeres kode. Micro:Bitten sender IKKE en event når den forlader vip tilstand og komme i vandret. Det kan i deducere ved at tilføje en timeout på input fra Micro:Bitten. Når der ikke kommer events så er den i ro i vandret position.

Se kode eksempel på GitHub – M1Se-Filterinput. Kør det og se hvad det er den kode gør. Den gør flere ting så læs kommentarerne i koden som det første.

```
ser = serial.Serial(timeout=1)
ser.baudrate = 115200
ser.port = "COM5"
```

```
test_print_1 = False
test_print_2 = True
```

```
light_events = ["N", "D", "S", "n", "d", "s"]
current_light_state = None
```

```
tilt_events = ["H", "V", "F", "T", "h", "v", "f", "t"]
current_tilt_state = None
```

```
relevant_sender_1 = "Knud-3"
relevant_sender_2 = "Knud 1"
```

```
def receive_char():
    microbitdata = str(ser.readline())
    if test_print_1: print(len(microbitdata), microbitdata)
    if len(microbitdata) > 3:
        return microbitdata[2], microbitdata[4:14]
    else:
        return None, None

def relevant_event(rc, who):
    if (who == relevant_sender_1) and (rc in light_events):
        return rc
    elif (who == relevant_sender_2) and (rc not in light_events):
        return rc
    else:
        return None
```

```
ser.open()

while True:
    rc, who = receive_char()
    if rc == None: # Nothing received - taht is a timeout on the serial port then
        if test_print_1: print("timeout - nothing received")
        current_tilt_state = None # not tilted as we have a timeout from MB reading.
    else: # Something received
        if test_print_1: print("received character:", rc, "from:", who, "String with the lenght", len(who))
        event = relevant_event(rc, who)
        if event != None:
            if event in tilt_events:
                if event == current_tilt_state:
                    if test_print_2: print("Do repeated tilt stuff if any", who, event)
                else:
                    tilts_all_since_start += 1
                    if event == "V" or event == "v":
                        tilts_left_since_start += 1
                    if test_print_2:
                        print("Do tilt stuff on first tilt event in specifik direction and set tilt state", who, event)
                        print("The code for tilt can then take current light state into account if you want", current_light_state)
                        print("No of unique - not repeated - tilts in any direction since programstart:", tilts_all_since_start)
                        print("No of unique - not repeated - Vv tilts since programstart:", tilts_left_since_start)
                        current_tilt_state = event
            elif event in light_events:
                if test_print_2: print("Do light stuff and remember current light state for use elsewhere", who, event)
                current_light_state = event
            else: # Other input than tilt and light
                if event == "A" or "a":
                    Aa_pressed_since_start += 1
                if test_print_2:
                    print("Do stuff on other input than tilt and light", who, event)
                    print("The code for other inputs can then take current light state into account if you want", current_light_state)
                    print("No of A and a presses since programstart:", Aa_pressed_since_start)
```

# LKM1: FAQ – Hvad gør man hvis man vil samle op på antal events

?

Vi vil gerne holde styr på hvor mange gange en vippe har vippet og vil derfor gerne tilføje en tæller eller flere som summerer op på hvor mange gange en kanp er trykket eller at der er vippet i en bestemt retning etc. Hvordan får man samlet det op

!

I skal lave nogle variable der bruges til at opsummere undervejs. De skal initialiseres til nul inden jeres while løkke, og så skal der lægges en til variablen hvr gang der er en relevant event som i vil holde styr på. I kan selvfølgelig så også clere den ved bestemte events hvis I ønsker det.

Til illustration/inspiration er der lavet 3 tællere i M1Se-Filterinput.py på Githubben. Se kode eksempel på GitHub – M1Se-Filterinput.

Kør det og se hvad det er den kode gør. Den gør flere ting så læs kommentarerne i koden som det første.

Kan I få programmet til at cleare (sætte til nul) ex. tilt tæller ved tryk på "b" i den kode?

```
ser = serial.Serial(timeout=1)
ser.baudrate = 115200
ser.port = "COM5"

test_print_1 = False
test_print_2 = True

light_events = ["N", "D", "S", "n", "d", "s"]
current_light_state = None

tilt_events = ["H", "V", "F", "T", "h", "v", "f", "t"]
current_tilt_state = None

relevant_sender_1 = "Knud-3"
relevant_sende_2 = "Knud 1"
```

```
def receive_char():
    microbitdata = str(ser.readline())
    if test_print_1: print(len(microbitdata), microbitdata)
    if len(microbitdata) > 3:
        return microbitdata[2], microbitdata[4:14]
    else:
        return None, None

def relevant_event(rc, who):
    if (who == relevant_sender_1) and (rc in light_events):
        return rc
    elif (who == relevant_sender_2) and (rc not in light_events):
        return rc
    else:
        return None
```

```
ser.open()

while True:
    rc, who = receive_char()
    if rc == None: # Nothing received - taht is a timeout on the serial port then
        if test_print_1: print("timeout - nothing received")
        current_tilt_state = None # not tilted as we have a timeout from MB reading.
    else: # Something received
        if test_print_1: print("received character:", rc, "from:", who, "String with the lenght", len(who))
        event = relevant_event(rc, who)
        if event != None:
            if event in tilt_events:
                if event == current_tilt_state:
                    if test_print_2: print("Do repeated tilt stuff if any", who, event)
                else:
                    tilts_all_since_start += 1
                    if event == "V" or event == "v":
                        tilts_left_since_start += 1
                    if test_print_2:
                        print("Do tilt stuff on first tilt event in specifik direction and set tilt state", who, event)
                        print("The code for tilt can then take current light state into accout if you want", current_light_state)
                        print("No of unique - not repeated - tilts in any direction since programstart:", tilts_all_since_start)
                        print("No of unique - not repeated - Vv tilts since programstart:", tilts_left_since_start)
                        current_tilt_state = event
            elif event in light_events:
                if test_print_2: print("Do light stuff and remember current light state for use elsewhere", who, event)
                current_light_state = event
            else: # Other input than tilt and light
                if event == "A" or "a":
                    Aa_pressed_since_start += 1
                if test_print_2:
                    print("Do stuff on other input than tilt and light", who, event)
                    print("The code for other inputs can then take current light state into accout if you want", current_light_state)
                    print("No of A and a presses since programstart:", Aa_pressed_since_start)
```