

# Micro:bit measures temperature which is sent to MatDeck and displayed in instrument widget

This example illustrates communications between MatDeck and micro:bit using a com port. The obtained results are displayed in instrument widget.

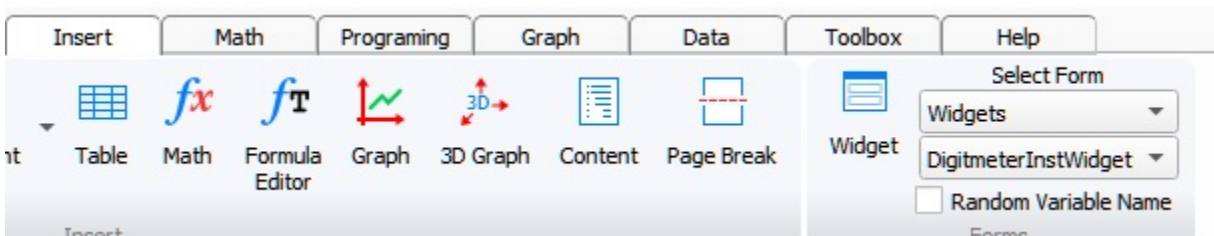
The document here contains the micro:bit Python program. The user can flash .HEX files onto a micro:bit directly from the document. They will need to highlight the whole Python block they would like to flash and click Deploy. If the micro:bit Python block has already been deployed to the micro:bit, you will not need to deploy it again to run it. The micro:bit should be connected to the PC. The receiver code is also in this MatDeck document

The micro:bit's processor contains a temperature sensor which can be used in your programs. It's a useful approximation of the temperature around the micro:bit.

- Flush the following code to your micro:bit (select all lines and click Deploy button from programming tab)

```
1 #py
2 from microbit import *
3
4 while True:
5     x = temperature()
6     print(x)
7     display.show(Image.YES)
8     sleep(250)
9     display.show(Image.NO)
10    sleep(250)
11
12 ###
```

The temperature read from the micro:bit unit is displayed by the virtual instrument in the canvas below. The instrument is added from **Insert - Select Form**.



It is related to the variable given in the code.

```
WGT2 := DigitmeterInstWidget("WGT2")
```



Temperature measured in Celsius

MatDeck can communicate and receive data from the micro:bit unit via com port. The micro Python code given above will cause the micro:bit to send temperature data via a com port. The data can then be displayed using Virtument. The required parameters for com port communications are:

- COM3
- Baud rate = 115200
- Data = 8 bits
- Parity = none
- Stop = 1 bit

```
13 handle := com_open("COM3,115200,N,8,1")
14 t := timer_create(250)
15 Temp := 0
16 counter := 10
17 on_event(t,microbit_read())
```

Here, we temperature sent from micro:bit.

```
18 microbit_read()
19 {
20   value := com_read(handle, 100)
21
22   if(size(value) == 4)
23   {
24     valustr := vec2str(value)
25     Temp = to_number(mid(valustr, 0,2))
26     set_widget_value(WGT2, Temp)
27   }
28   counter -= 1
29   if(counter == 0)
30   {
31     com_close(handle)
32     timer_delete(t)
33   }
34 }
```