LCCS 6

Microbit Project

Patrick Brennan

Mini Microbit Project

LCCS 6

Contents

Part 1

{Part 1a} Preparing the Microbit to send Temperature data over a Serial USB port on your computer {MakeCode}

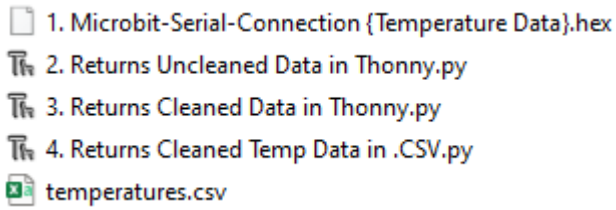
{Part 1b} Writing a Python program to read in the data from the Microbit and display in Thonny

Folder Setup

1. In your Documents folder create a new folder called: **Microbit-To-CSV**.
2. When your exercise is finished and working, you will have these 5 files.

Note: File. **4. Returns Cleaned Temp Data in .csv.py** contains ALL the code.

Note: File **temperatures.csv** is automatically created by Thonny. You don't need to make it.



Note: You already have the first THREE files from another class.

You are beginning at **4. Returns Cleaned Temp Data in .csv.py**

3. When completed and working, drag the whole folder called: **Microbit-To-CSV** to your OneDrive as part of your project material.

Serial Microbit Firebase Connection

What is this exercise about?

Uploading data FROM a Microbit up through a serial port and displaying in Thonny using Python

Note: **A microbit does not automatically assume that you are connecting it to a USB port. We have to explicitly tell it so. This is even though a USB port and a microbit communicate using a Serial connection.**

Part 1 {Capturing and Uploading the data}

USB Connection to Emulate SERIAL Communication

Step 1: From the Serial toolbox, drag out a **serial** block and stick it in the **on start** block.

a. Redirect both connections to USB

b. Set the **baud rate** to 115200 {rate at which the 'bits' are transferred over serial connection}

The **baud rate** is the rate at which information is transferred in a communication channel in bits per second.

Step 2: Drag a **serial write number** into the **forever block.**

a. Add a **Temperature** block to it.

b. Add a **serial write line** block beneath the previous block.

We will put speech marks here with a space. This is insert a blank line between our temperature readings to make it more readable.

c. The **show number** and **temperature** blocks show the temperature on the Microbit.

d. The pause of 5000ms or 5 seconds only allow the temperature to be read into the Python program in Thonny every 5 seconds.

e. Download this code to the Microbit.

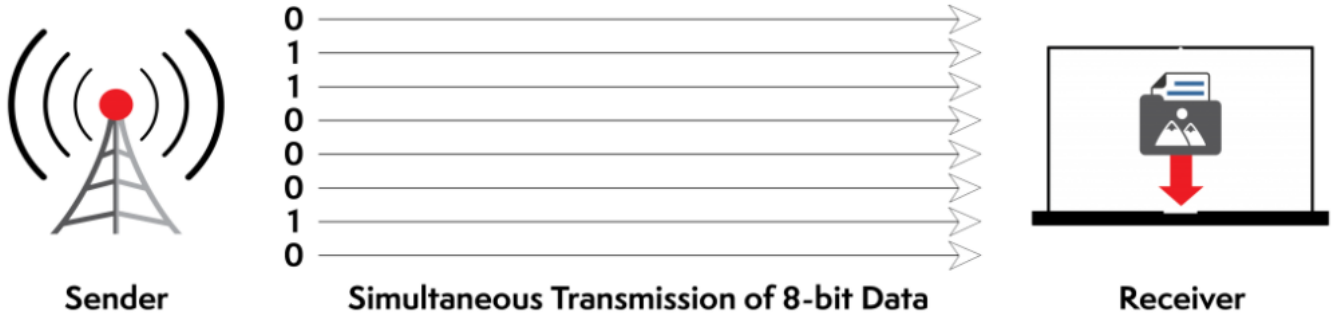
When we transfer this MakeCode 'code' to our Microbit by dragging it, we are **emulating** or **creating a virtual port connection** that allows the USB interface to function as a **traditional** {physical port} serial port connection.

A little about Ports

Serial Port: It is called a serial port because it 'serializes' the data. That means that it sends it one bit at a time. It uses only **one** wire to do this.

Note: A bit is only sent if the previous data bit has been received.

Parallel Port: A parallel port is **eight** times faster because it uses **eight** wires to transfer data.



Universal Serial Bus (USB) is a SERIAL Communication Interface:

USB 2.0	Hi-Speed USB	**480 Mbps**
USB 3.2 Gen 1	USB 3.0 USB 3.1 Gen 1 SuperSpeed	**5 Gbps**
USB 3.2 Gen 2	USB 3.1 USB 3.1 Gen 2 SuperSpeed+ SuperSpeed	**10Gbps**

1. We use a 'serial over USB' connection because it allows data to travel **one bit at a time**, similar to a traditional serial port {which was physical port}.

It uses **flow control** to prevent the microbit **or** computer from being overwhelmed when transferring continuous data, such as temperature readings from a sensor.

Note: USB is already a serial communication port {like the Microbit}. What we're doing is **emulating** or **creating a virtual port connection** that allows the USB interface to function as a **traditional** {physical port} serial port connection.

Note: In the code below, you will have to find the correct port that your microbit is connected to.

You do this by opening **Command Prompt** on your machine and typing: **mode**

{Part 1b}

Python program to read in the data from the Microbit and display in Thonny

Please note that this will import 'uncleaned data'.

```
import serial   # Serial Library
import time     # Time Library

ser = serial.Serial() # We use the serial library to create a serial connection
ser.baudrate = 115200 # Set the baud rate to the same as that on the Microbit
ser.port = "com5"  # This virtual port changes. Use Cmd prompt and type: mode to get correct one
ser.open() # Open the port for communication.

while True:
    microbitData = str(ser.readline())  # Incoming data is a 'byte' object. Coverted to 'string' for easy manipulation
    temperature = microbitData
    temperature =str(temperature) # Converts temperature to an integer

    print(temperature)

ser.close() # Good Practice but actually doesn't do anything.
```

Return Data in Thonny Realtime

Please note that this will import 'cleaned data'.

```
import serial   # Serial Library
import time     # Time Library

ser = serial.Serial() # We use the serial library to create a serial connection
ser.baudrate = 115200 # Set the baud rate to the same as that on the Microbit
ser.port = "com5"  # This virtual port changes. Use Cmd prompt and type: mode to get correct one
ser.open() # Open the port for communication.

while True:

    microbitData = str(ser.readline())  # Incoming data is a 'byte' object. Coverted to 'string' for easy manipulation
    temperature = microbitData

    temperature = microbitData[2:]  # Gets rid of first 2 characters
    temperature = temperature.replace(' ','') # Gets rid blank of spaces
    temperature = temperature.replace("'",'') # Two "" with a ' in middle gets rid of the ' at the end

    temperature = temperature.replace('\\r\\n','') # Gets rid of the \r\n but has to use \\r\\n

    temperature =int(temperature) # int if you want to do calculations with this data. Could be str.

    print(temperature)

ser.close() # Good Practice but actually doesn't do anything.
```

Return Cleaned Data in a .csv file

```
import csv
import serial
import time

ser = serial.Serial()
ser.baudrate = 115200
ser.port = "com5"
ser.open()

header = ['Temperature']
# Open the CSV file for writing
with open('temperatures.csv', 'w', newline='') as csvfile:
    db = csv.writer(csvfile)
    db.writerow(header)

    while True:
        microbitData = str(ser.readline())
        temperature = microbitData
        temperature = microbitData[2:]
        temperature = temperature.replace(' ','')
        temperature = temperature.replace("'","')
        temperature = temperature.replace('\\r\\n','')
        temperature =int(temperature) # Not Necessary Always but can't do calculations without int conversion
        print(temperature)

        # Write the temperature to the CSV file
        db.writerow([temperature, temperature * 2]) # Square Brackets to create list with single element
        # Data written to .csv must be in a sequence {An iterable object that can be looped through}.
        csvfile.flush()  # Flush the data to ensure that it is written to the file

time.sleep(5)
ser.close()
```