BBC micro:bit
Bit:Commander Text Entry

Introduction

It's easy to get text displayed on the micro:bit. The only barrier to using micro:bits to send text messages is the difficulty in being able to enter text using only the built-in inputs. With the Bit:Commander, that process is much easier. It is not quite the keyboard experience in full but is getting much more workable.

This program lets the user enter a message to scroll across the display. The comments in the code explain how you use the components on the board to select text.

The Program

There is room for improvement, but it is getting near to usable.

```
# System for text entry using Bit:Commander
from microbit import *

# charset is a list of ASCII codes in our character set
charset = [i for i in range(65,91)] # upper case letters
charset += [k for k in range(48,58)] # digits
charset += [l for l in range(32,48)] # punctuation and symbols

# chars is a list of characters in our character set
chars = [chr(i) for i in charset]

# read the buttons and use binary 4 bits to represent
# the button states
def get_btns():
    pattern = 0
    for i,p in enumerate([pin12,pin15,pin14,pin16]):
        pattern += p.read_digital() << i
    return pattern

# creates a list of 2 bit values representing buttons states
# 00 - button not pressed, 01 - button pressed
# 10 - button released, 11 - button held
def get_evts(prev,current):
    return [((prev >> i & 1)<<1) + (current >> i & 1) for i in range(4)]


# function to get a string from the user
# red button confirms letter
# blue button confirms string
# green button to view message
# joystick Y to navigate letters
def get_string():
    letter = 0
    tick = 1
    display.show(chars[letter])
```

BBC Microbit

Collapse All Expand All

+ Block Editor - The Basics
+ Block Editor - Components
+ Kodu - micro:bit Worlds
+ JavaScript Blocks
+ JavaScript Blocks - Exercises
+ Blocks - Bit:Bot
+ Blocks - Bit:Commander
+ MicroPython - Starting Off
+ MicroPython - Examples
+ MicroPython - Components
+ MicroPython - Breakout Boards
+ MicroPython - Exercises
+ MicroPython - Pi Accessories
+ MicroPython - Bit:Bot
- MicroPython - Bit:Commander
  - Bit:Commander
  - The Joystick
  - The Neopixels
  - The Potentiometer
  - The Pushbuttons
  - The Buzzer
  - Evasion Game
  - Light's Out Game
  - Simon Game
  - Bit:Bot/Robot Controller
  - Text Entry
  - Unicorn Commander
+ MicroPython - Projects
+ MicroPython - Visual Basic
+ Other - Odds & Ends

```
    last = 0
    usertext = ""
    while True:
        btns = get_btns()
        e = get_evts(last, btns)
        if e[0]==2:
            usertext += chars[letter]
            sleep(250)
        elif e[1]==2:
            return usertext
        elif e[2]==2:
            display.scroll(usertext)
        else:
            a = pin2.read_analog()
            if a<150:
                letter -=1
                sleep(250)
            elif a>750:
                letter +=1
                sleep(250)
            # prevent scrolling beyond list length
            letter = max(0, min(letter, len(chars)-1))
            display.show(chars[letter])
        # 'follow' the button states
        last = btns
        if tick>0:
            display.show(chars[letter])
        else:
            display.clear()
        tick *= -1
        sleep(50)

msg = "Press stick"
while True:
    a = pin8.read_digital()
    if a:
        msg = get_string()
    display.scroll(msg)
```

It is relatively easy to add the radio sending code to make it possible to send the message to another micro:bit. There could also be some more enhancements to the letter selection process.