Make Learn Design

Contents

Introduction 2

Microbit and Microsoft Block editor 2

Lesson 1: Learning the basics with Micro:bit 2

Getting started / Introduction 3

Next coding challenges 6

Challenge 2: Hello world: shake it 6

Challenge 3: Temperature scroller 7

Challenge 4: Basic button pressing 7

Challenge 5 : Basic button pressing with sprites 8

Challenge 6: Loops 12

Challenge 7: Variables 16

MicroPython and the Microbit 18

Getting started 18

Challenge 1 Hello world, hello mum! 19

Challenge 2 using buttons 21

Challenge 3 Random Name scroller 22

Challenge 4 Shake it random nickname generator 23

Challenge 5 shake it, dice roller 24

Challenge 6 Shake the bit, display a random picture 25

Micropython going beyond the basics 27

On the Micro:bit 27

Functions and pictures and Neopixels 27

Moody storm trooper : temp determined 29

On the Micro:bit 29

V2 with realistic temps 32

On the Micro:bit 32

Disco Storm Trooper 34

On the Micro:bit 34
Neon Axe MB Micropython 35
On the Micro:bit 36
Radio activated lights on Micro:bit V1(Raspberry Pi or PC/ Mac compatible at time of writing 08/16)........ 37
On the Micro:bit 38
Minecraft Pi and Micro:bit TNT roulette 40
On the Micro:bit 41
Minecraft Pi and Micro:bit messages. 44
Minecraft Pi and Micro:bit : 'Random serial bounce' 48
On the Micro:bit 48
V2 Radio Control Minecraft Jump V2........ 55

Introduction

Microbit and Microsoft Block editor

Lesson 1: Learning the basics with Micro:bit

What you will learn?

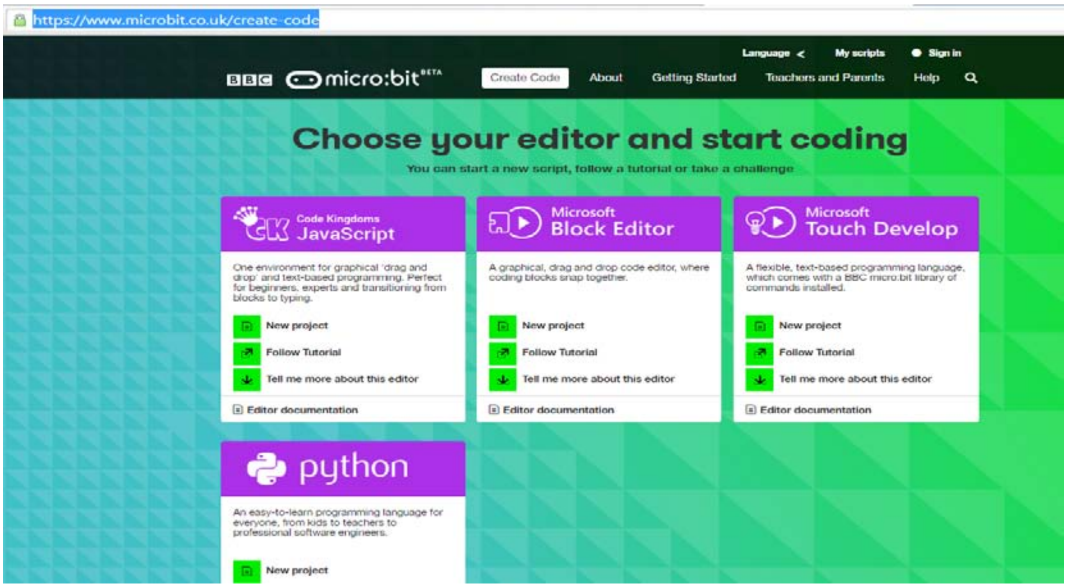
- How to create sprites
- Getting used to the drag and drop language / environment
- Using conditionals and basic loops
- How to scroll text across the screen
- How to use the accelerometer to trigger an message on screen
- Display the CPU temperature
- Basic button pressing
- Using all buttons and accelerometer to trigger different sprites/ faces

Getting started / Introduction

Getting started with Micro:bit is very easy using the Microsoft touch drag and drop environment which can be accessed from here:

https://www.microbit.co.uk/create-code

The BBC website:



Choose your editor and start coding

You can start a new script, follow a tutorial or take a challenge

Code Kingdoms JavaScript: One environment for graphical 'drag and drop' and text-based programming. Perfect for beginners, experts and transitioning from blocks to typing.

Microsoft Block Editor: A graphical, drag and drop code editor, where coding blocks snap together.

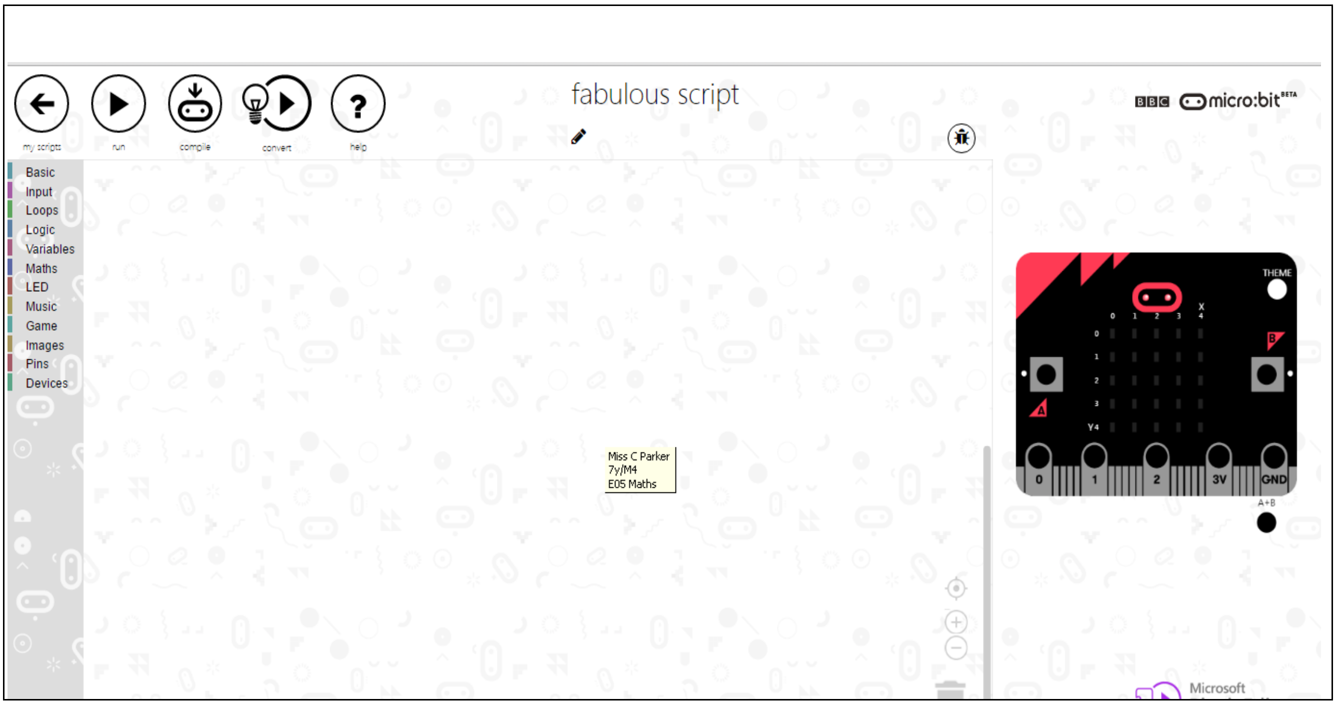
Microsoft Touch Develop: A flexible, text-based programming language, which comes with a BBC micro:bit library of commands installed.

python: An easy-to-learn programming language for everyone, from kids to teachers to professional software engineers.

We will start off by using the Microsoft Block editor by clicking create a new project.

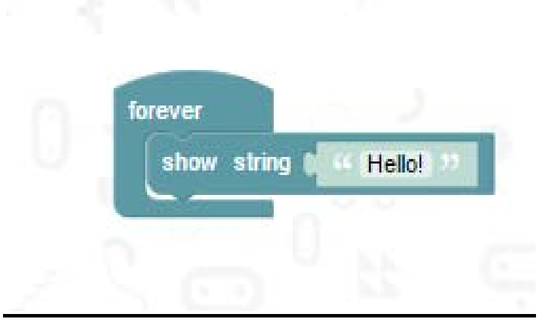
Steps

1. Plug your Micro:bit into the PC with the USB cable provided
2. Go onto the site listed above, using the Microsoft Block editor section click create new project.
3. You will see a screen that looks like this:

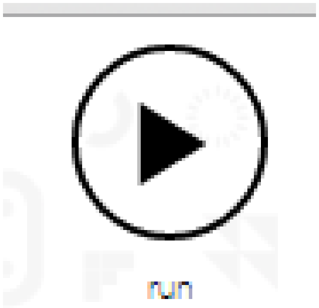
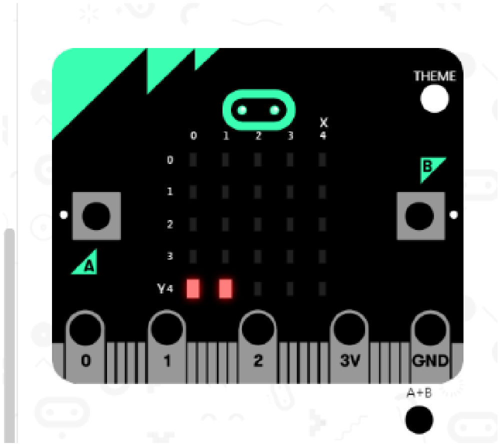


4. We will be using the ‘basic’ tab on the left side menu to create our hello world program.

5. **Lets get stuck into the code**

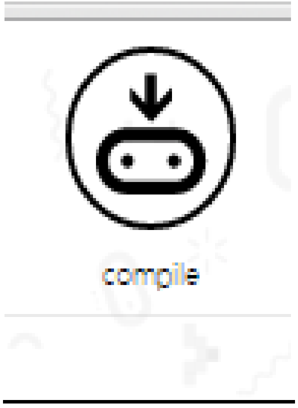
Code	**Help / explanation**
 forever show string “Hello!”	Permanently while Micro:bit is plugged in Scroll text “Hello”

6. To preview this then click on the run button, this will then preview the program on screen emulator(see below:)

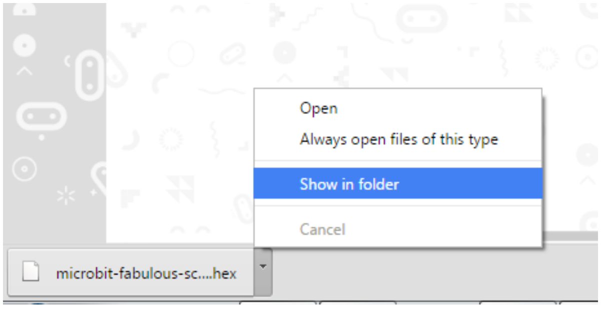
Button	Emulator example
	

Next step when you have completed this is to then get it working your Micro:bit.

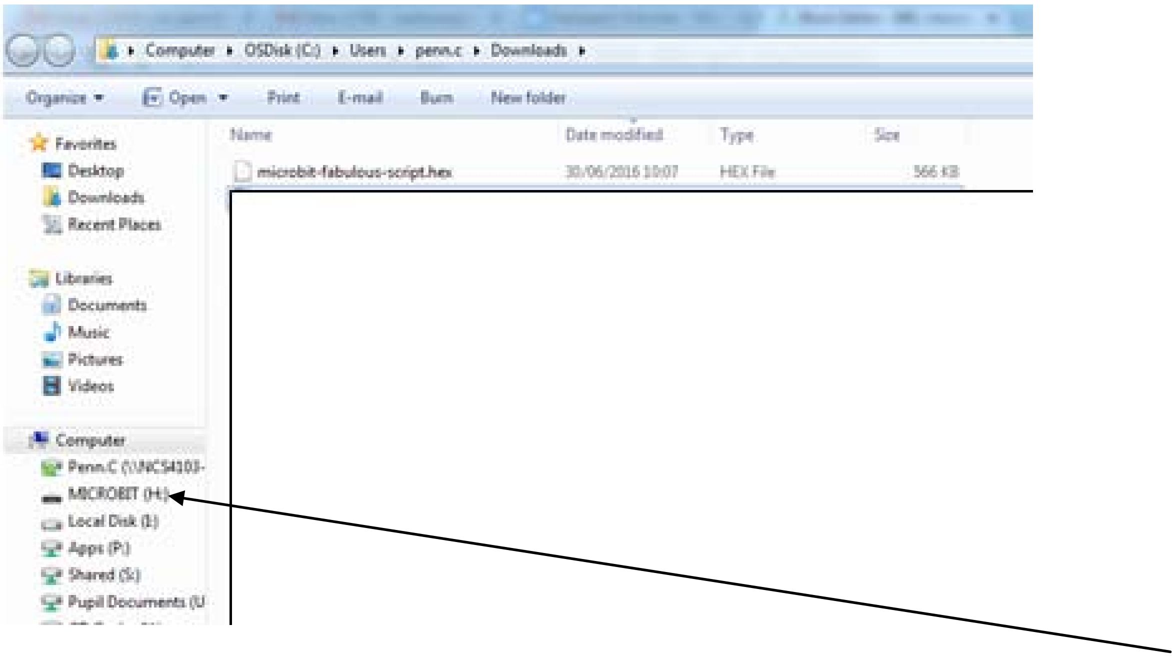
To do this click on the compile button:



Then the code Hex file will appear in your downloads folder , to access it right click and go ‘show in folder’:



Then go to ‘start’, ‘computer’ and ‘downloads’ your hex file should saved there like so:



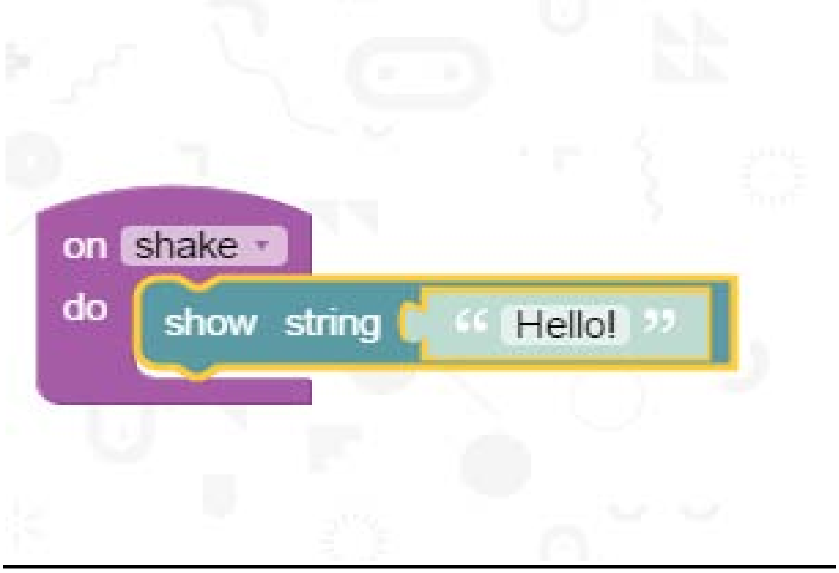
Finally you can drag and drop the hex file onto the Micro:bit which should appear like a USB memory stick.

You will then have to wait for 10-15 seconds while the file transfers (the yellow light located on the back of the Micro:bit will flash at this time), your Micro:bit program should now work.

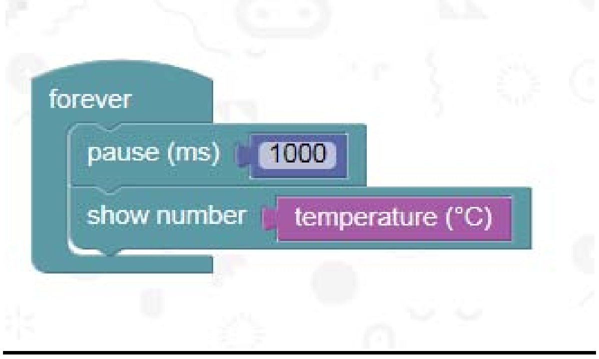
Next coding challenges

Challenge 2: Hello world: shake it

<u>**Code**</u>	<u>**Help / explanation**</u>

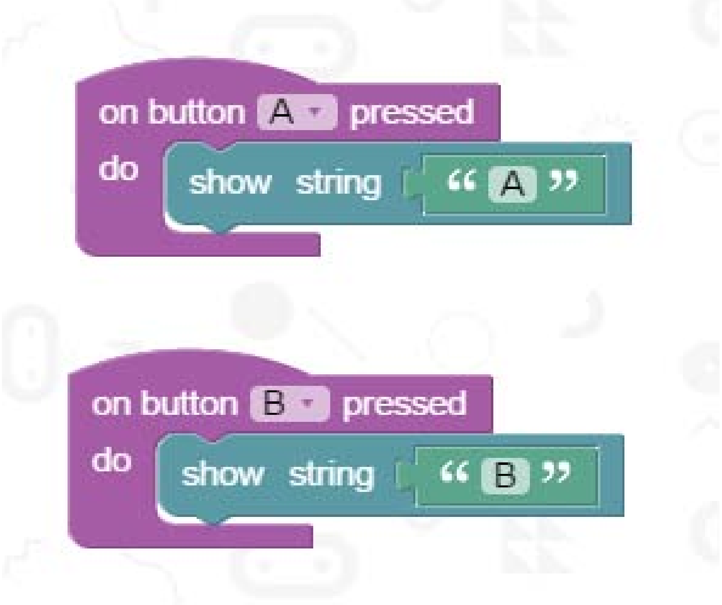
 on shake do show string “ Hello! ”	When Micro:bit shaken Show text “Hello”

Challenge 3: Temperature scroller

Code	**Help / explanation**
 forever pause (ms) 1000 show number temperature (°C)	Permanently loop Wait 1000 milliseconds (1 second) Display number of current CPU temperature

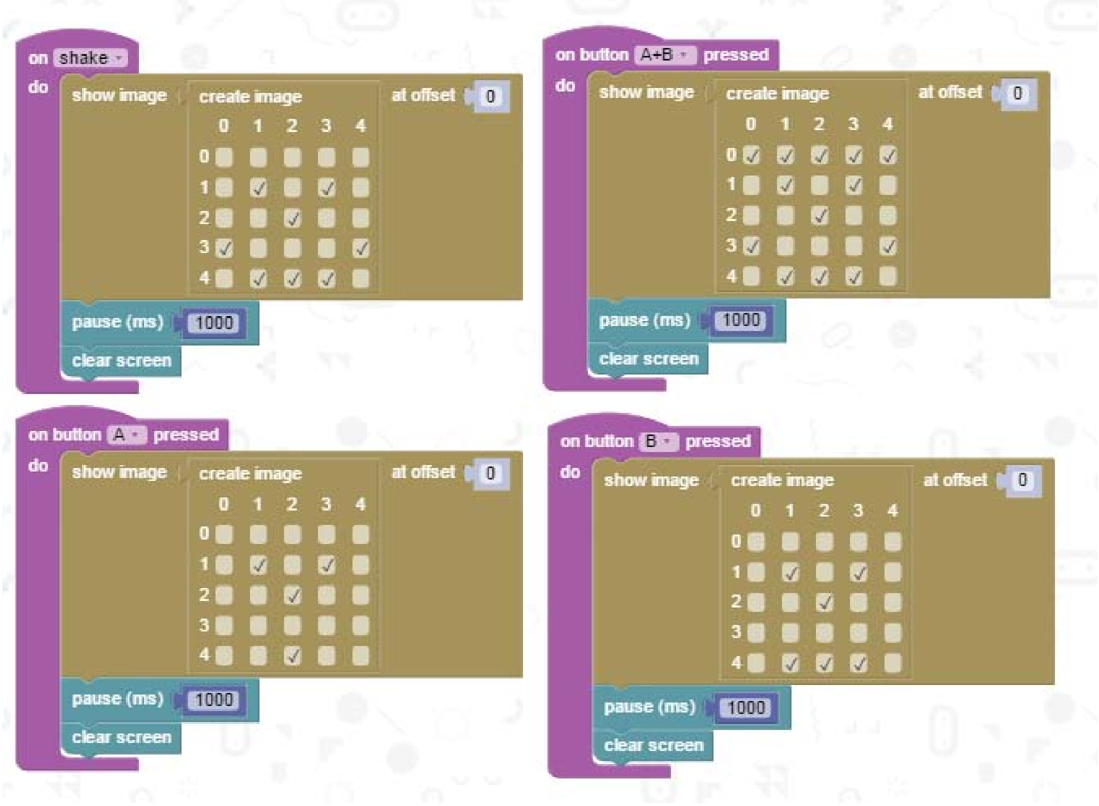
Challenge 4: Basic button pressing

Code	**Help / explanation**

 on button A pressed do show string "A" on button B pressed do show string "B"	**If** A button pressed **Then** Display Text "A" **If** B button pressed **Then** Display Text "B"

Challenge 5 : Basic button pressing with sprites

Code



```
on shake
do  show image  create image          at offset 0
                  0 1 2 3 4
                0 □ □ □ □ □
                1 □ ✓ □ ✓ □
                2 □ □ ✓ □ □
                3 ✓ □ □ □ ✓
                4 □ ✓ ✓ ✓ □
    pause (ms) 1000
    clear screen
```

```
on button A+B pressed
do  show image  create image          at offset 0
                  0 1 2 3 4
                0 ✓ ✓ ✓ ✓ ✓
                1 □ ✓ □ ✓ □
                2 □ □ ✓ □ □
                3 ✓ □ □ □ ✓
                4 □ ✓ ✓ ✓ □
    pause (ms) 1000
    clear screen
```

```
on button A pressed
do  show image  create image          at offset 0
                  0 1 2 3 4
                0 □ □ □ □ □
                1 □ ✓ □ ✓ □
                2 □ □ ✓ □ □
                3 □ □ □ □ □
                4 □ □ ✓ □ □
    pause (ms) 1000
    clear screen
```

```
on button B pressed
do  show image  create image          at offset 0
                  0 1 2 3 4
                0 □ □ □ □ □
                1 □ ✓ □ ✓ □
                2 □ □ ✓ □ □
                3 □ □ □ □ □
                4 □ ✓ ✓ ✓ □
    pause (ms) 1000
    clear screen
```

Help / notes

```
If A button pressed Then
    Display Image of pondering face
    Wait for 1000 milliseconds
    Clear screen ready for next instruction to be received

If B button pressed Then
    Display Image of sad face
    Wait for 1000 milliseconds
    Clear screen ready for next instruction to be received

If A button And B pressed Then
    Display Image of smiley face with hair
    Wait for 1000 milliseconds
    Clear screen ready for next instruction to be received

If Mico:bit has been shaken Then
    Display Image of smiley face
    Wait for 1000 milliseconds
    Clear screen ready for next instruction to be received
```

Well done you have now completed your first set of Micro:bit challenges

Extension

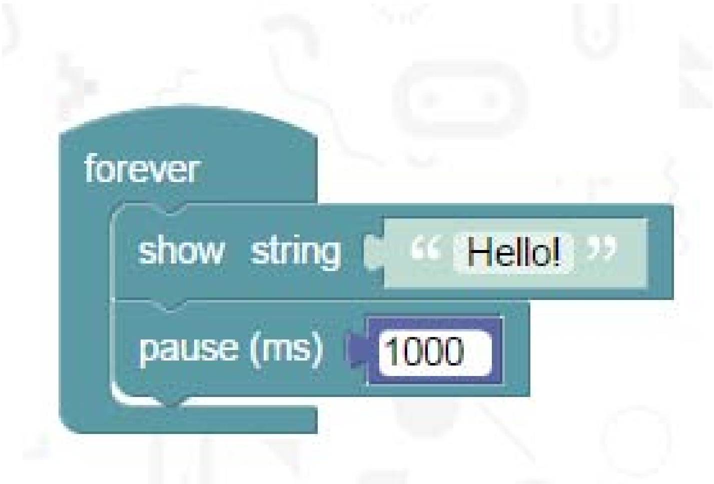
Try remixing challenge 5 by creating different images and or scrolling different messages

Challenge 6: Loops

Earlier you used a forever loop to complete a task over and over again using a loop. It is the same as the forever loop used in Scratch if you have used this before.

Code

V1 Hello .... wait!   Add a delay

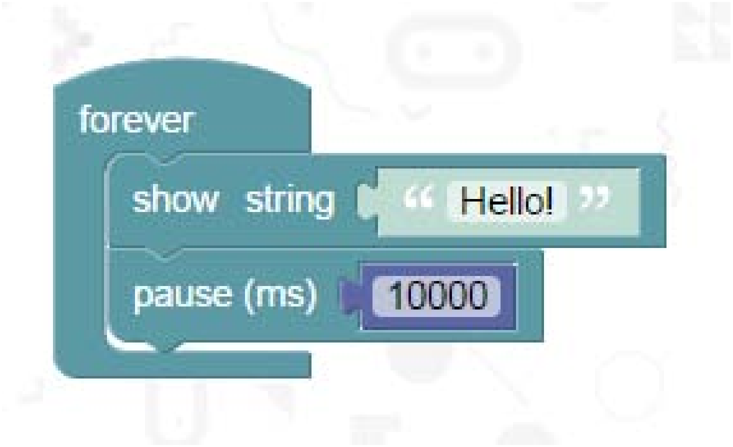


```
forever
    show string "Hello!"
    pause (ms) 1000
```

n.b. 1000 milliseconds = 1 second

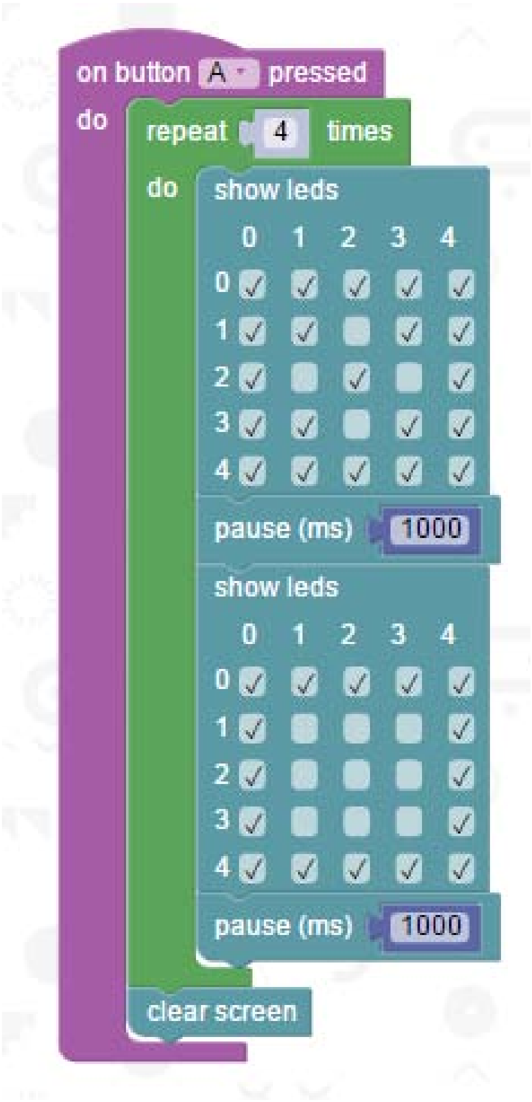
Add a delay so that you can see what is going on in your program

V2 Hello .... wait even longer!   Add a delay



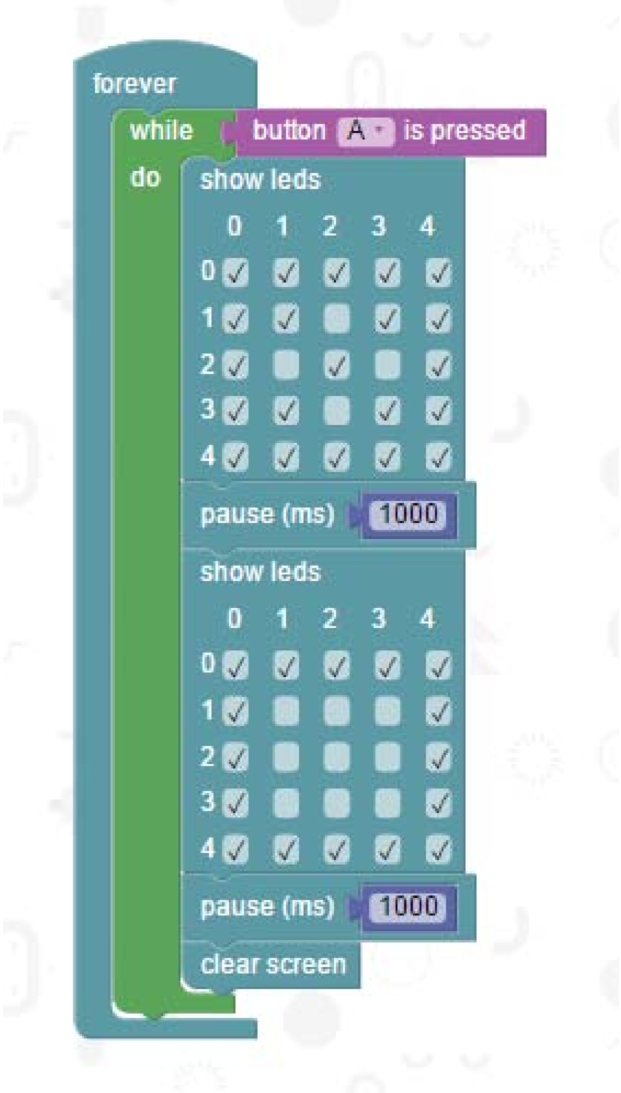
```
forever
    show string "Hello!"
    pause (ms) 10000
```

V3 Press a button animation (count controlled loop)



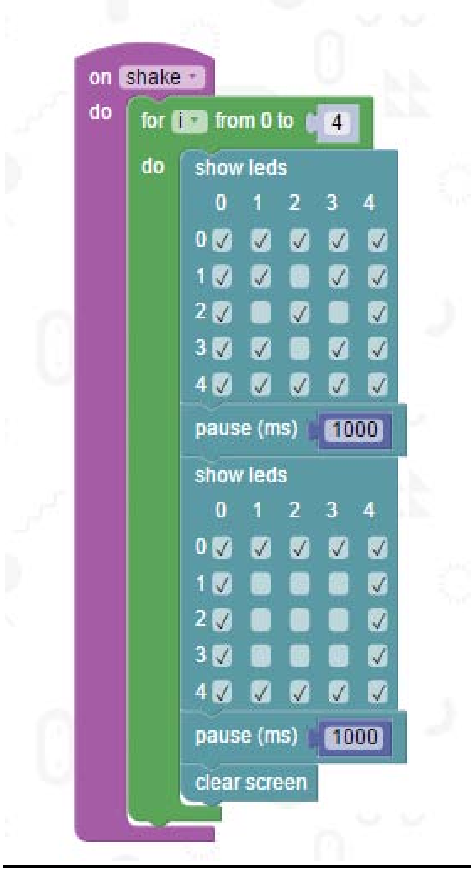
```
on button A pressed
do  repeat 4 times
    do  show leds
              0  1  2  3  4
            0 ☑  ☑  ☑  ☑  ☑
            1 ☑  ☑  □  ☑  ☑
            2 ☑  □  ☑  □  ☑
            3 ☑  ☑  □  ☑  ☑
            4 ☑  ☑  ☑  ☑  ☑
        pause (ms) 1000
        show leds
              0  1  2  3  4
            0 ☑  ☑  ☑  ☑  ☑
            1 ☑  □  □  □  ☑
            2 ☑  □  □  □  ☑
            3 ☑  □  □  □  ☑
            4 ☑  ☑  ☑  ☑  ☑
        pause (ms) 1000
    clear screen
```

V4 Forever check a button



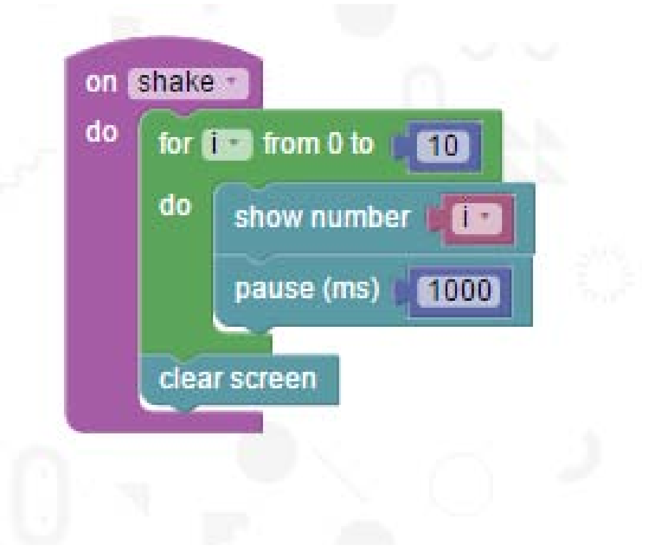
```
forever
  while  button A ▾ is pressed
  do
    show leds
          0  1  2  3  4
        0 ☑  ☑  ☑  ☑  ☑
        1 ☑  ☑  □  ☑  ☑
        2 ☑  □  ☑  □  ☑
        3 ☑  ☑  □  ☑  ☑
        4 ☑  ☑  ☑  ☑  ☑
    pause (ms) 1000
    show leds
          0  1  2  3  4
        0 ☑  ☑  ☑  ☑  ☑
        1 ☑  □  □  □  ☑
        2 ☑  □  □  □  ☑
        3 ☑  □  □  □  ☑
        4 ☑  ☑  ☑  ☑  ☑
    pause (ms) 1000
    clear screen
```

V5 Shake it (For loop)



```
on shake
do  for i from 0 to 4
    do  show leds
              0 1 2 3 4
            0 ✓ ✓ ✓ ✓ ✓
            1 ✓ ✓ □ ✓ ✓
            2 ✓ □ ✓ □ ✓
            3 ✓ ✓ □ ✓ ✓
            4 ✓ ✓ ✓ ✓ ✓
        pause (ms) 1000
        show leds
              0 1 2 3 4
            0 ✓ ✓ ✓ ✓ ✓
            1 ✓ □ □ □ ✓
            2 ✓ □ □ □ ✓
            3 ✓ □ □ □ ✓
            4 ✓ ✓ ✓ ✓ ✓
        pause (ms) 1000
        clear screen
```

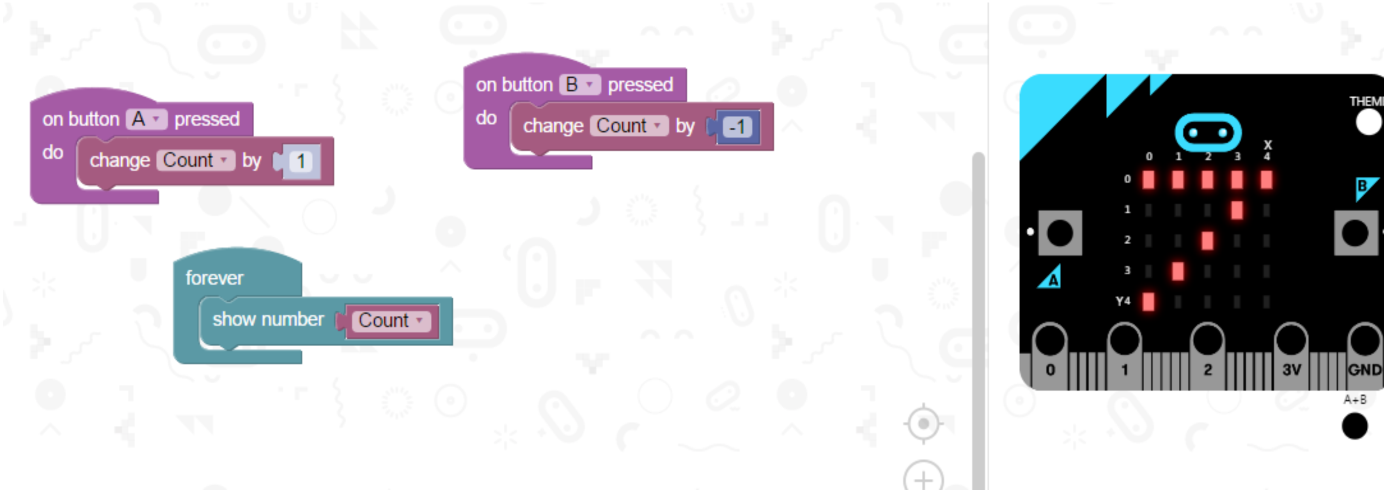
V6 For loop counter



```
on shake
do  for i from 0 to 10
    do  show number i
        pause (ms) 1000
    clear screen
```

Challenge 7: Variables

7.1 Button clicker

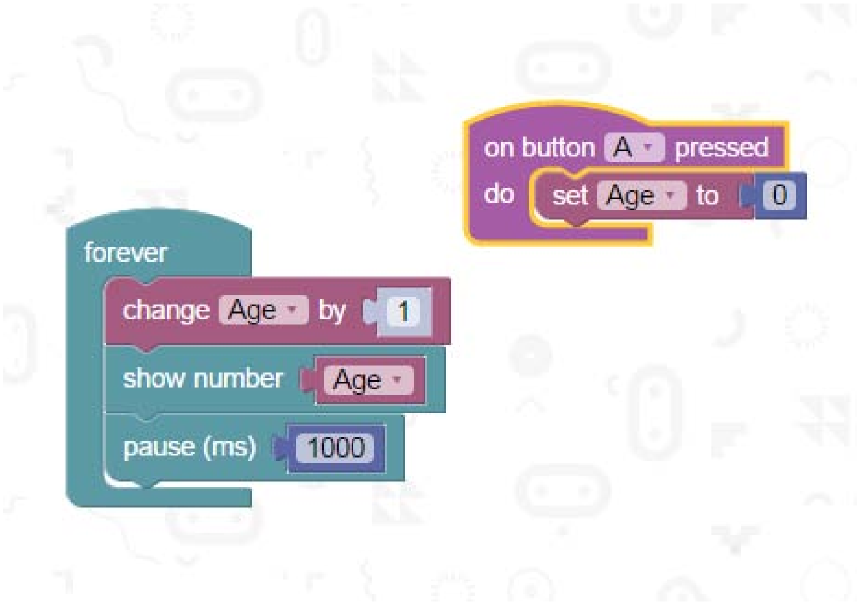


```
on button A pressed
do  change Count by 1
```

```
on button B pressed
do  change Count by -1
```

```
forever
    show number Count
```

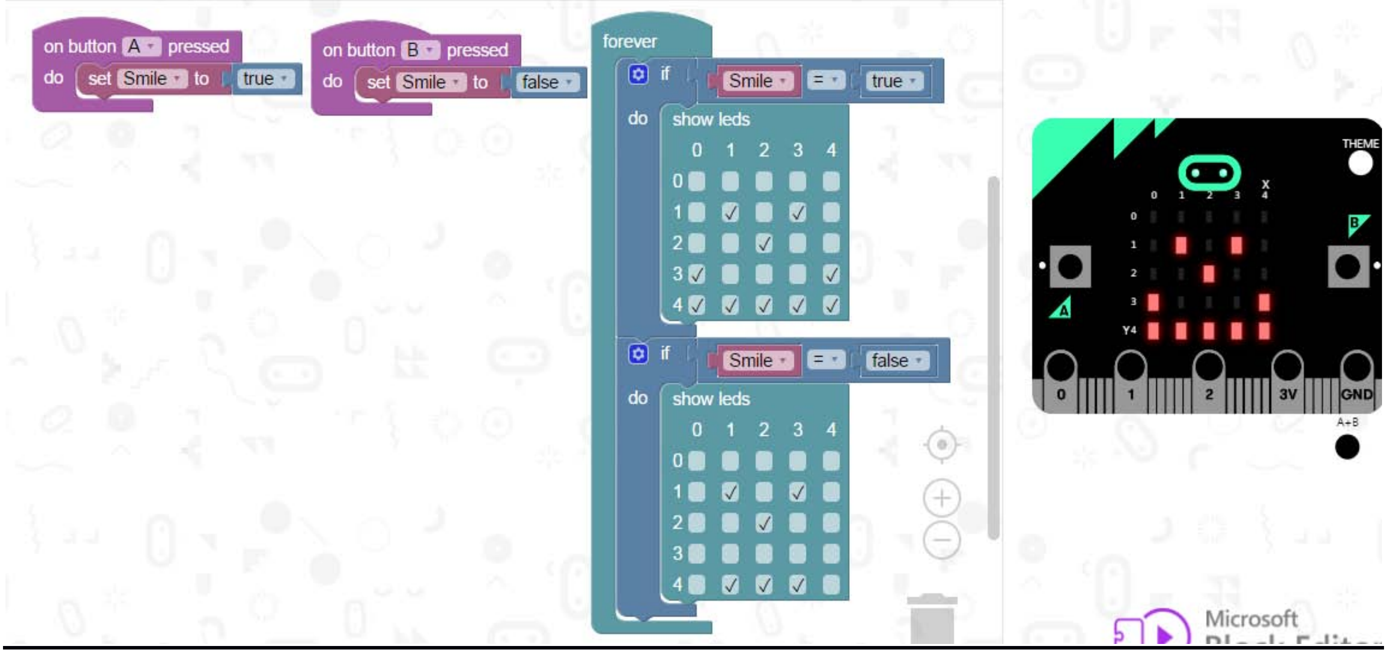
7.2 Basic stopwatch



```
on button A pressed
do  set Age to 0
```

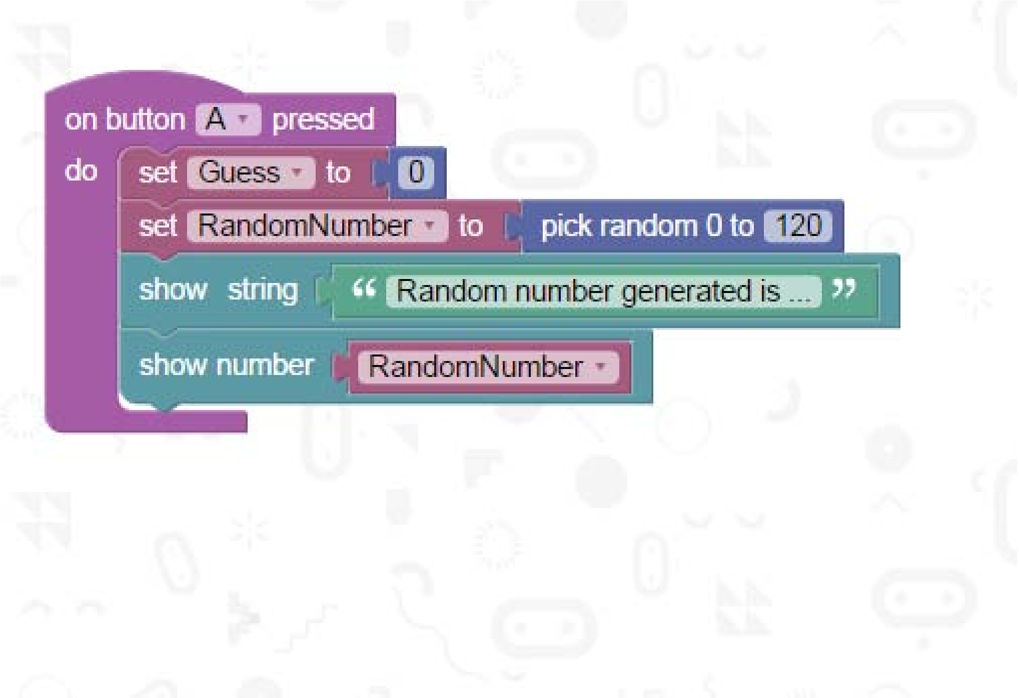
```
forever
    change Age by 1
    show number Age
    pause (ms) 1000
```

7.3 Boolean smile



```
on button A pressed
do  set Smile to true

on button B pressed
do  set Smile to false

forever
    if Smile = true
    do  show leds
            0 1 2 3 4
        0   □ □ □ □ □
        1   □ ✓ □ ✓ □
        2   □ □ ✓ □ □
        3   ✓ □ □ □ ✓
        4   ✓ ✓ ✓ ✓ ✓
    if Smile = false
    do  show leds
            0 1 2 3 4
        0   □ □ □ □ □
        1   □ ✓ □ ✓ □
        2   □ □ ✓ □ □
        3   □ □ □ □ □
        4   □ ✓ ✓ ✓ □
```

7.4 Random number generator



```
on button A pressed
do  set Guess to 0
    set RandomNumber to pick random 0 to 120
    show string "Random number generated is ..."
    show number RandomNumber
```

MicroPython and the Microbit

Getting started

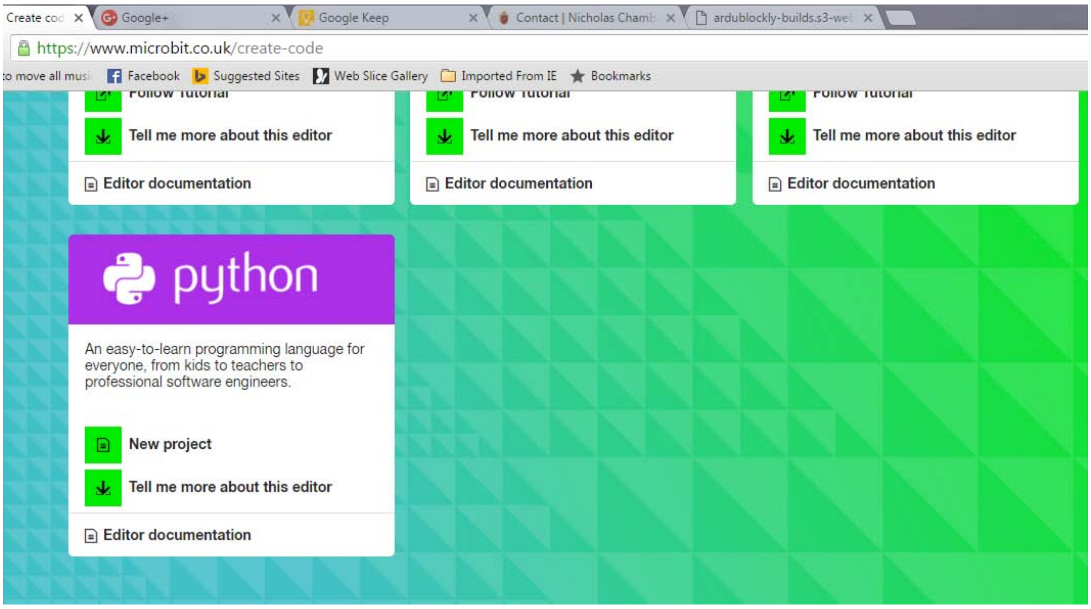
MicroPython is a text based programming language unlike the blocks you have been just using.

You have a few options when coding with MicroPython and the Micro:bit these are:

- The web based https://www.microbit.co.uk/create-code editor for MicroPython called MU
- The downloadable version which can be installed on PC, Mac and Linux based machines including Raspberry Pi(more on this later). You can download the version that suits you from here. http://ardublockly-builds.s3-website-us-west-2.amazonaws.com/?prefix=microbit
- Please note for the Raspberry Pi version you will need to download the Binary file (personally I found it easier to get it working. More on this later.)

In honesty the majority of you will use the BBC site in your class room. So we will start here.

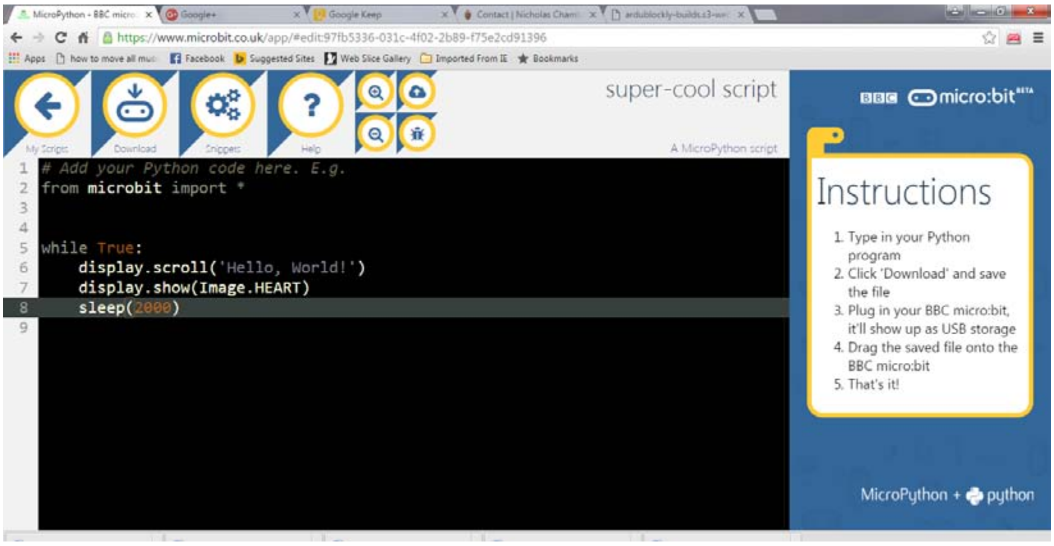
The Micro:bit website looks like this:


https://www.microbit.co.uk/create-code
Follow Tutorial
Tell me more about this editor
Editor documentation
python
New project
Tell me more about this editor
Editor documentation

An easy-to-learn programming language for everyone, from kids to teachers to professional software engineers.

You will need to select new project.

The interface will load up and look like this...



```
# Add your Python code here. E.g.
from microbit import *


while True:
    display.scroll('Hello, World!')
    display.show(Image.HEART)
    sleep(2000)
```

Instructions

1. Type in your Python program
2. Click 'Download' and save the file
3. Plug in your BBC micro:bit, it'll show up as USB storage
4. Drag the saved file onto the BBC micro:bit
5. That's it!

Step 1 plug in your Micro:bit to the USB port. Lets create a our first text based program for Micro:bit.

Challenge 1 Hello world, hello mum!

This does the basic hello world and says a hello to a few more people along the way using a basic list data structure to store the names, try it out:

Key concepts used here are:

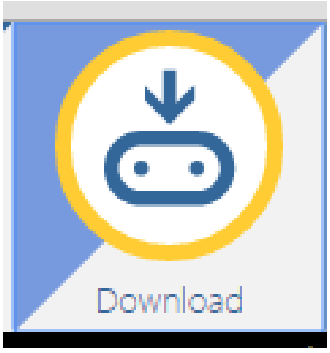
- Lists
- Iteration
- Index

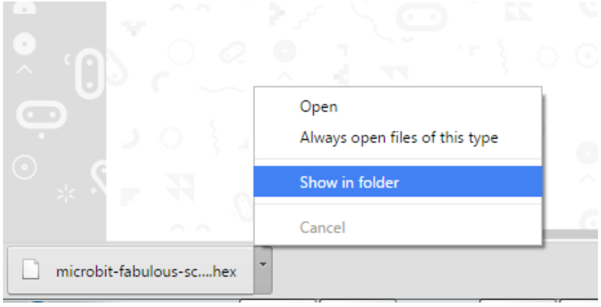
- Count controlled loops
- Accessing values stored in a list
- joining together string values
- Casting

Code

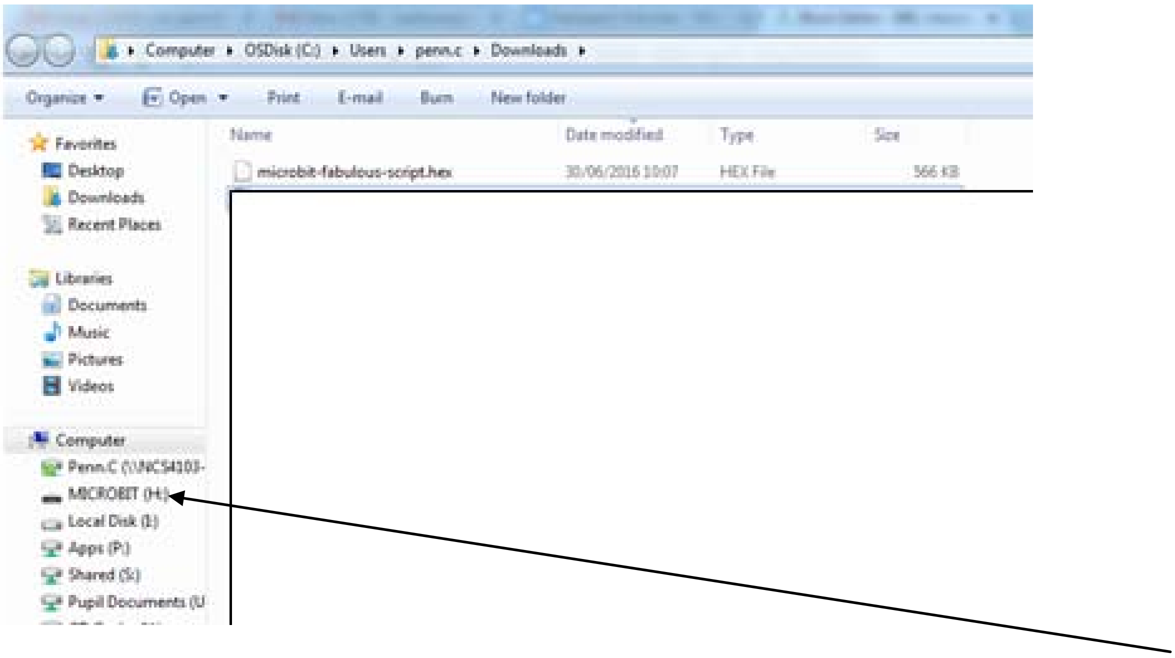
```
import microbit

namesList = ["Nanny June","Daddy","Mum"] #list of 3 names

index = 0

microbit.display.scroll("Hello World") #message
while index <=2:
	msg = "Hello "+str(namesList[index])
	microbit.display.scroll(msg)
	index = index +1
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.





Then go to ‘start’, ‘computer’ and ‘downloads’ your hex file should be saved there like so:



Name	Date modified	Type	Size
microbit-fabulous-script.hex	30/06/2016 10:07	HEX File	566 KB

Finally you can drag and drop the hex file onto the Micro:bit which should appear like a USB memory stick.

You will then have to wait for 10-15 seconds while the file transfers (the yellow light located on the back of the Micro:bit will flash at this time), your Micro:bit program should now work.

Challenge 2 using buttons

This challenge is a basic example of getting the Micro:bit to respond to buttons ‘A’ and ‘B’ being pressed.

The second example uses the 'A' and 'B' buttons to print out messages to screen.

Testing out using the buttons try this code by creating a test program:

Code:

```
import microbit

while True:

    if microbit.button_a.is_pressed():
        microbit.display.scroll("This is a ...")

    if microbit.button_b.is_pressed():
        microbit.display.scroll("....test program")
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously in challenge 1.

Challenge 3 Random Name scroller

This third program makes use of two lists and uses two lists to display a random name made up of a randomly selected first name then a randomly selected second name.

Key concepts used here are:

- Multiple lists
- Infinite loops
- Accessing values stored in two lists

```
import microbit

import random

RandomFirstName = ["Steve","Shannon","Jenny"]

RandomSecondName = ["Debank","Green","Penny","Smith"]
```

```
while True:

    microbit.display.scroll(random.choice(RandomFirstName)

    microbit.sleep(1000)#wait 1 sec

    microbit.display.scroll(random.choice(RandomSecondName))

    microbit.sleep(1000)#wait 1 sec
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Challenge 4 Shake it random nickname generator

The fourth example is a random nickname generator based on adjectives for body type and random names It uses the majority of the coding concepts previously looked at except it introduces the shake.

Code:

```
from microbit import *

import random

NameList = ["Paul","Dave","Gem","Rachel","Chris"]

BodyTypeAdjectiveList = ["Sturdy","Bullnecked","Gangling","Heavyset","Lanky","Musclebound"]

while True:

    if accelerometer.was_gesture('shake'):# if shaken then
```

```
        RandomNickname = random.choice(BodyTypeAdjectiveList)+" "+random.choice(NameList)

        display.scroll(RandomNickname)
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Challenge 5 shake it, dice roller

This simple dice simulation introduces the accelerometer and how to use it to randomly simulate a 6 sided dice.

Code

```
from microbit import *
import random

DiceNumbers = [1,2,3,4,5,6]#List of 6 possible numbers

while True:
    if accelerometer.was_gesture('shake'):# if shaken then
        msg = "You rolled.. "+str(random.choice(DiceNumbers))
        display.scroll(msg)#show the text
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Challenge 6 Shake the bit, display a random picture

This basically allows you to shake the microbit and this will randomly display one of the library of images. N.B. I have only implemented a few to give a brief idea :)

Concepts covered:

- Lists
- Functions
- Loops
- Accelerometer
- Random library *Conditional statements

Code

```
from microbit import *
import random

"""
List of  possible pictures not sure if it is exhaustive, stored as string so they can be stored in a list
this process is called casting.
"""
PicNamesList =
[str(Image.SAD),str(Image.HEART),str(Image.MEH),str(Image.RABBIT),str(Image.COW)]


"""
I have created a function that groups the code and makes the final program much cleaner
it basically:
*imports the list as a parameter
```

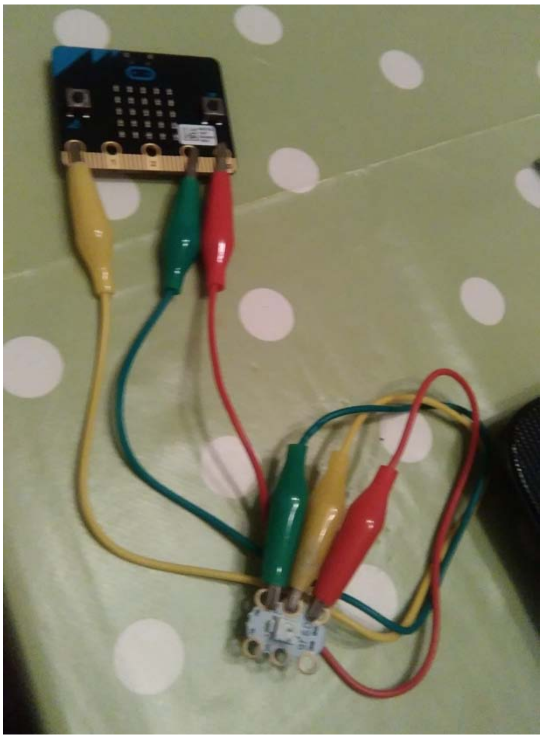
```
*creates a temp variable which stores the string representation of the image randomly selected
*then uses if and elif statements to check which image it should display on microbit
"""

def checkWhichImageIAm(PicNamesList):
    chosenImage = random.choice(PicNamesList)
    if chosenImage == str(Image.SAD):
        display.show(Image.SAD)
    elif chosenImage == str(Image.HEART):
        display.show(Image.HEART)
    elif chosenImage == str(Image.MEH):
        display.show(Image.MEH)
    elif chosenImage == str(Image.RABBIT):
        display.show(Image.RABBIT)
    elif chosenImage == str(Image.COW):
        display.show(Image.COW)

while True:
    if accelerometer.was_gesture('shake'):# if shaken then
        checkWhichImageIAm(PicNamesList)# run check which image am I function
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Micropython going beyond the basics

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

Functions and pictures and Neopixels

```
from microbit import *
import neopixel

np = neopixel.NeoPixel(pin0,1)

def Angry():
    np[0] = (255,0,0)#Red
    np.show()
    display.show(Image.ANGRY)
    sleep(3000)
    np.clear()

def Happy():
    np[0] = (0,255,0)#Green
    np.show()
    display.show(Image.HAPPY)
    sleep(3000)
    np.clear()

def Meh():
    np[0] = (255,69,0)#Yellow
    np.show()
    display.show(Image.MEH)
    sleep(3000)
    np.clear()

while True:
    Happy()
```

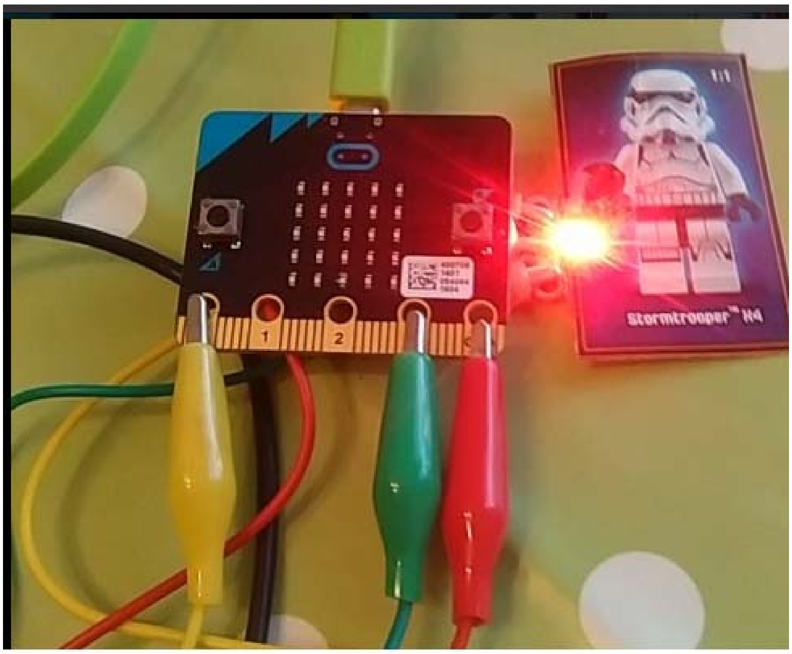
```
Angry()

Meh()
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your 'download' folder.

Now download your code as a hex file as you have done previously.

Moody storm trooper : temp determined

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

Code

```
# Add your Python code here. E.g.

from microbit import *

import neopixel
```

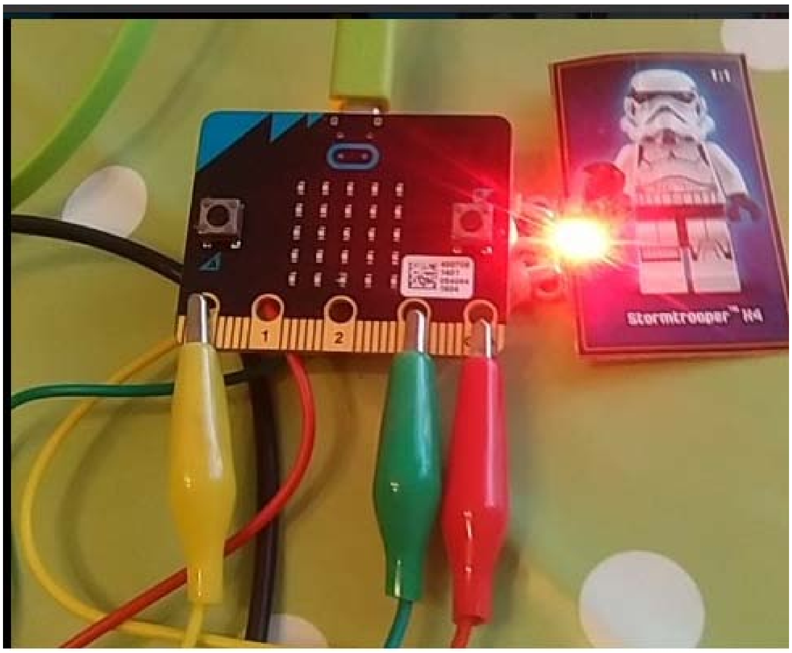
```
np = neopixel.NeoPixel(pin0,1)

def Angry():
    np[0] = (255,0,0)#Red
    np.show()
    display.show(Image.ANGRY)
    sleep(3000)
    np.clear()

def Happy():
    np[0] = (0,255,0)#Green
    np.show()
    display.show(Image.HAPPY)
    sleep(3000)
    np.clear()

def Meh():
    np[0] = (255,69,0)#Yellow
    np.show()
    display.show(Image.MEH)
    sleep(3000)
    np.clear()

while True:
    temp  = temperature()
    if temp <= 27:#happy at body temp
        Happy()
    elif temp > 27 or temp <=29:#above body temp but ok
        Meh()#ok
    elif temp > 29 or temp <= 31:
        Angry()
    else:
        display.show(Image.SKULL)#
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

V2 with realistic temps

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

Code

```
from microbit import *

import neopixel


np = neopixel.NeoPixel(pin0,1)


def Angry():

    np[0] = (255,0,0)#Red

    np.show()

    display.show(Image.ANGRY)

    sleep(3000)

    np.clear()
```

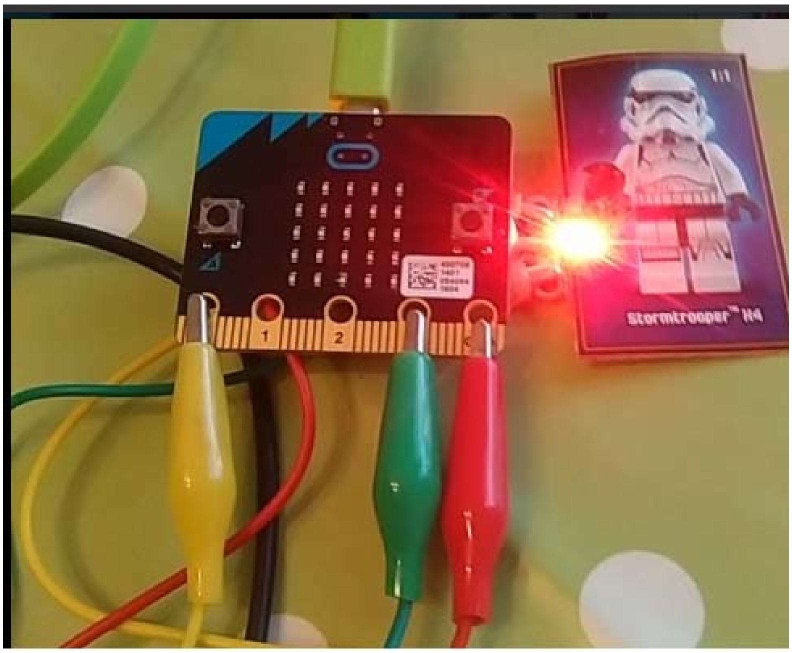
```
def Happy():
    np[0] = (0,255,0)#Green
    np.show()
    display.show(Image.HAPPY)
    sleep(3000)
    np.clear()


def Meh():
    np[0] = (255,69,0)#Yellow
    np.show()
    display.show(Image.MEH)
    sleep(3000)
    np.clear()


while True:
    temp  = temperature()
    if temp <= 25:#happy at body temp
        Meh()#ok / cool
    elif temp >= 25 or temp <=50:#above body temp but ok
        Happy()#ok / coo
    elif temp > 50 or temp <= 75:#super hot
        Angry()
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Disco Storm Trooper

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

Random int

Based on read the docs demo

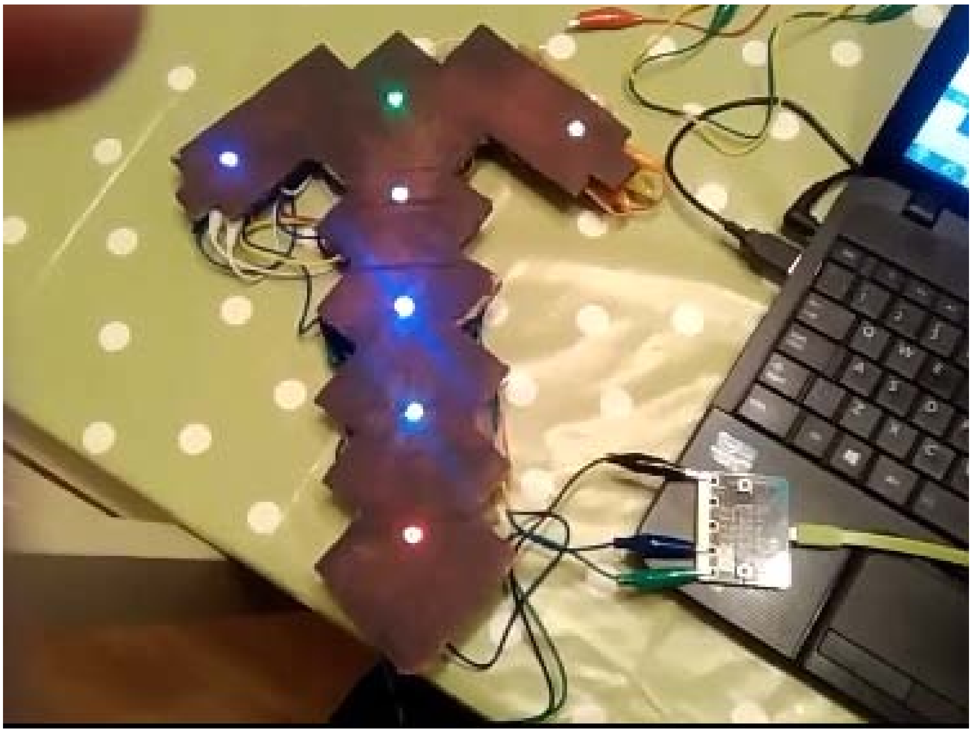
```
from microbit import *
import neopixel
import random

np = neopixel.NeoPixel(pin0, 1)
while True:
    red = random.randint(0, 60)
    green = random.randint(0, 60)
    blue = random.randint(0, 60)
     # Assign the current LED a random red, green and blue value between 0 and 60
    np[0] = (red, green, blue)
     # Display the current pixel data on the Neopixel strip
    np.show()
    sleep(100)
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Neon Axe MB Micropython

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

Code

```
"""
    neopixel_random.py

    Repeatedly displays random colours onto the LED strip.
    This example requires a strip of 8 Neopixels (WS2812) connected to pin0.

"""
from microbit import *
```

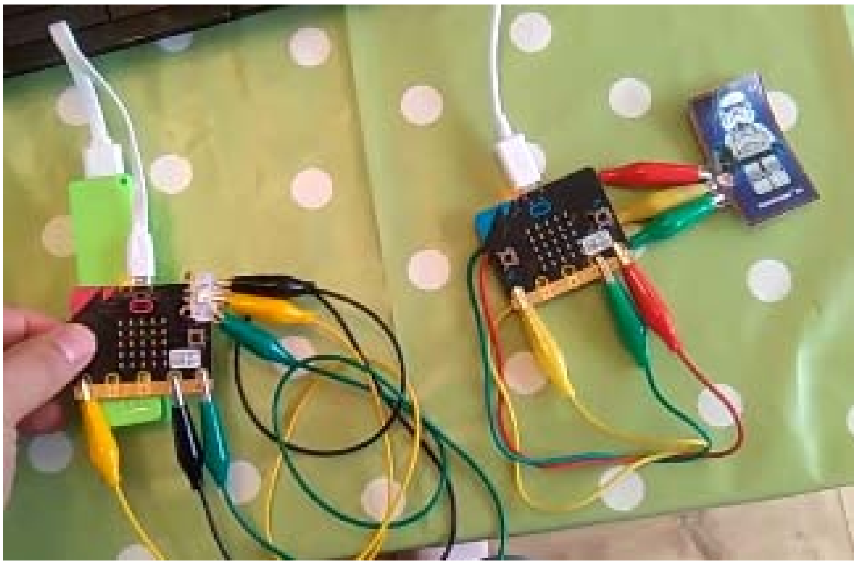
```
import neopixel
from random import randint


# Setup the Neopixel strip on pin0 with a length of 8 pixels
np = neopixel.NeoPixel(pin0, 7)


while True:
    #Iterate over each LED in the strip


    for pixel_id in range(0, len(np)):
        red = randint(0, 60)
        green = randint(0, 60)
        blue = randint(0, 60)


        # Assign the current LED a random red, green and blue value between 0 and 60
        np[pixel_id] = (red, green, blue)


        # Display the current pixel data on the Neopixel strip
        np.show()
        sleep(100)
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Radio activated lights on Micro:bit V1(Raspberry Pi or PC/ Mac compatible at time of writing 08/16)

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

You will need to code the Micro:bit with MU either on a PC/ MAC or Linux machine. At the time of writing the BBC website will not allow the radio module to work. Thus to be able to compile the code you will need to install the MU IDE from **here**

This program uses the radio on the Micro:bit to send messages to trigger Neopixels on other Microbits. The radio code here is taken from the demo by N Toll on the Micro:bit read the docs website.

You will need at least two Micro:bits in order to see it working, both Micro:bits will need the same code loaded onto it.

Code to go onto the Micro:bit

```
from microbit import *

import neopixel



np = neopixel.NeoPixel(pin0,1)



import radio

import random

#import neopixel

#from microbit import display, Image, button_a, sleep
```

```
#np = neopixel.NeoPixel(pin0,1)

def Angry():
    np[0] = (255,0,0)#Red
    np.show()
    display.show(Image.ANGRY)
    sleep(3000)
    display.clear()
    np.clear()

def Happy():
    np[0] = (0,255,0)#Green
    np.show()
    display.show(Image.HAPPY)
    sleep(3000)
    display.clear()
    np.clear()

def Meh():
    np[0] = (255,69,0)#Yellow
    np.show()
    display.show(Image.MEH)
    sleep(3000)
    display.clear()
    np.clear()
```

```
radio.on()
while True:
    FeelingsList = ['Angry','Meh','Happy']
    # Button A sends a "flash" message.
    emo = random.choice(FeelingsList)
    if button_a.was_pressed():
        radio.send(emo)  # a-ha
    # Read any incoming messages.
    incoming = radio.receive()
    if incoming == 'Meh':
        Meh()
    elif incoming == 'Happy':
        Happy()
    elif incoming == 'Angry':
        Angry()
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Minecraft Pi and Micro:bit TNT roulette

Picture of Micro:bit set setup

  Minecraft - Pi edition Button pressed = ,True,+ Block ID = ,46, + ,Active= 1	On the Micro:bit

Instructions

You will need to have a Raspberry Pi and Micro:bit for this to work. You will need to install MU on the Raspberry Pi

This program uses the radio on the Micro:bit to send messages to trigger Neopixels on other Microbits. The radio code here is taken from the demo by N Toll on the Micro:bit read the docs website.

Python code to run in python on the Raspberry Pi

```
"""

Code written by Martin O'Hanlon in the following blog post:

http://www.stuffaboutcode.com/2016/03/microbit-get-data-from-usb.html

"""

import serial

from mcpi.minecraft import Minecraft

from time import sleep

from mcpi import block as block

import random




PORT = "/dev/ttyACM0"

BAUD = 115200
```

```
s = serial.Serial(PORT)

s.baudrate = BAUD

s.parity   = serial.PARITY_NONE

s.databits = serial.EIGHTBITS

s.stopbits = serial.STOPBITS_ONE

#read the first line and flush any bad data

s.readline()


def read_microbit_data():

    #read a line from the microbit,

    data = s.readline()

    #split the microbit data into x, y, z, a, b

    data_s = data.rstrip().split(" ")

    a = True if data_s[0] == "True" else False

    b = True if data_s[1] == "True" else False

    BlockID = int(data_s[2])

    Active = int(data_s[3])

    return a,b,BlockID,Active

mc = Minecraft.create()

try:

    playerPos = mc.player.getTilePos()

    while True:

        a,b,BlockID, Active = read_microbit_data()

        if a == True:

            pos = mc.player.getPos()

            msg = "Button pressed = ",str(a),"+ Block ID = ",BlockID," + ", "Active=",Active

            mc.postToChat(msg)

            mc.setBlock(pos.x,pos.y,pos.z,BlockID,Active)
```

```
finally:

    sleep(1)

    s.close()
```

Code to go on Micro:bit

Written by David Whale and Martin O’Hanlon edited by Chris Penn

```
from microbit import*

import random


REFRESH = 500


BlockID = 46

ActiveValues = [0,1] # 0 = off


def get_data():

    a, b = button_a.was_pressed(), button_b.was_pressed()

    Active  = random.choice(ActiveValues)

    print(a, b, BlockID,Active)


def run():

    while True:

        sleep(REFRESH)

        get_data()


display.show('M')

run()
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Minecraft Pi and Micro:bit messages.

Picture of Micro:bit set setup

Python code to run in python on the Raspberry Pi

```
import mc_microbit as m

import time
```

```
"""
Code written by Martin O'Hanlon in the following blog post:
http://www.stuffaboutcode.com/2016/03/microbit-get-data-from-usb.html
"""
import serial
from mcpi.minecraft import Minecraft
from time import sleep
from mcpi import block as block
import random



PORT = "/dev/ttyACM0"
BAUD = 115200

s = serial.Serial(PORT)
s.baudrate = BAUD
s.parity   = serial.PARITY_NONE
s.databits = serial.EIGHTBITS
s.stopbits = serial.STOPBITS_ONE
#read the first line and flush any bad data
s.readline()

def read_microbit_data():
    #read a line from the microbit,
    data = s.readline()
    #split the microbit data into x, y, z, a, b
    data_s = data.rstrip().split(" ")
    a = True if data_s[0] == "True" else False
    b = True if data_s[1] == "True" else False

    Message = data_s[2]

    return a,b,Message


mc = Minecraft.create()


try:

    playerPos = mc.player.getTilePos()

    m.build()#build mb

    while True:

        a,b,Message = read_microbit_data()

        if a == True

            pos = mc.player.getPos()

            mc.postToChat(Message)

            m.microbit.display.scroll(Message)


finally:

    sleep(1)

    s.close()
```

Code to go on Micro:bit

Written by David Whale and Martin O’Hanlon edited by Chris Penn

```
from microbit import*

import random


REFRESH = 500
```

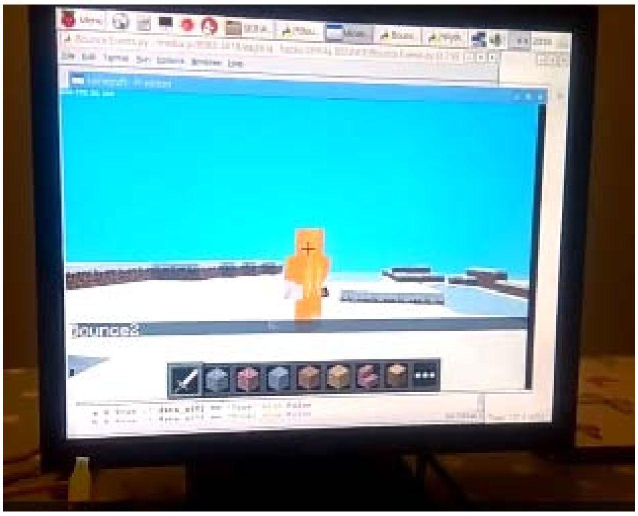
```
BlockID = 46

ActiveValues = [0,1] # 0 = off

Message = "Hello"

def get_data():

    #x, y, z = accelerometer.get_x(), accelerometer.get_y(), accelerometer.get_z()

    a, b = button_a.was_pressed(), button_b.was_pressed()


    print(a, b, Message)


def run():

    while True:

        sleep(REFRESH)

        get_data()


     #if button_a.was_pressed():

        display.scroll(Message)

run()
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Minecraft Pi and Micro:bit : ‘Random serial bounce’

Picture of Micro:bit set setup

	On the Micro:bit

Instructions

You will need to have a Raspberry Pi and Micro:bit for this to work. You will need to install MU on the Raspberry Pi

This program uses the radio on the Micro:bit to send messages to trigger Neopixels on other Microbits. The radio code here is taken from the demo by N Toll on the Micro:bit read the docs website.

Python code to run in python on the Raspberry Pi

```
import time


"""
Code written by Martin O'Hanlon in the following blog post:
http://www.stuffaboutcode.com/2016/03/microbit-get-data-from-usb.html
"""
import serial
from mcpi.minecraft import Minecraft
from time import sleep
from mcpi import block as block
import random



PORT = "/dev/ttyACM0"
BAUD = 115200


s = serial.Serial(PORT)
s.baudrate = BAUD
s.parity   = serial.PARITY_NONE
s.databits = serial.EIGHTBITS
s.stopbits = serial.STOPBITS_ONE
#read the first line and flush any bad data
s.readline()
```

```
def read_microbit_data():
    #read a line from the microbit,
    data = s.readline()
    #split the microbit data into x, y, z, a, b
    data_s = data.rstrip().split(" ")
    a = True if data_s[0] == "True" else False
    b = True if data_s[1] == "True" else False
    Message = data_s[2]
    yValue = data_s[3]
    return a,b,Message,yValue


mc = Minecraft.create()
try:
    playerPos = mc.player.getTilePos()
    while True:
        a,b,Message,yValue = read_microbit_data()
        if a == True:
            pos = mc.player.getPos()


            mc.player.setPos(pos.x,yValue,pos.z)
            msg = Message+""+yValue
            mc.postToChat(msg)
finally:
    sleep(1)
    s.close()
```

Code to go on Micro:bit

Written by David Whale and Martin O’Hanlon edited by Chris Penn

```
from microbit import*

import random


REFRESH = 500

Message = "Bounce"

def get_data():

    yValue = random.randint(2,20)

    a, b = button_a.was_pressed(), button_b.was_pressed()


    print(a, b, Message, yValue )


def run():

    while True:

        sleep(REFRESH)

        get_data()

        if button_a.is_pressed():

            display.scroll(Message)

run()
```

Now, read your code over to avoid any mistakes. Then click on download and the hex file will be sent to your ‘download’ folder.

Now download your code as a hex file as you have done previously.

Modify the Raspberry Code to teleport Steve when he jumps(modified code in red)

Picture of teleportation modification

Bounce19
Bounce7

```
import time


"""
Code written by Martin O'Hanlon in the following blog post:
http://www.stuffaboutcode.com/2016/03/microbit-get-data-from-usb.html
"""
import serial
from mcpi.minecraft import Minecraft
from time import sleep
from mcpi import block as block
import random



PORT = "/dev/ttyACM0"
BAUD = 115200


s = serial.Serial(PORT)
s.baudrate = BAUD
s.parity   = serial.PARITY_NONE
s.databits = serial.EIGHTBITS
```

```
s.stopbits = serial.STOPBITS_ONE
#read the first line and flush any bad data
s.readline()

def read_microbit_data():
    #read a line from the microbit,
    data = s.readline()
    #split the microbit data into x, y, z, a, b
    data_s = data.rstrip().split(" ")
    a = True if data_s[0] == "True" else False
    b = True if data_s[1] == "True" else False
    Message = data_s[2]
    yValue = data_s[3]
    return a,b,Message,yValue

mc = Minecraft.create()
try:
    playerPos = mc.player.getTilePos()
    while True:
        a,b,Message,yValue = read_microbit_data()
        if a == True:
            pos = mc.player.getPos()
            #change this to teleport


            mc.player.setPos(random.randint(-100,100), yValue,random.randint(-100,100))
            msg = Message+""+yValue
            mc.postToChat(msg)
```

```
finally:
    sleep(1)
    s.close()
```

Digital Pet v1

```
from microbit import *
#import time

PiggyHappy = Image("00000\n"
              "09990\n"
              "00000\n"
              "90009\n"
              "99999")

PiggySad = Image("00000\n"
              "09990\n"
              "00000\n"
              "00000\n"
              "09990")

Hunger = 20

while True:

    #if
    display.show(PiggyHappy)
    if button_b.is_pressed():
        Hunger = Hunger +button_a.get_presses()
    if Hunger <5:
        display.show(PiggySad)



    sleep(60000)
    Hunger = Hunger - 1
#display.clear()
```

V2 Radio Control Minecraft Jump V2

To go on micro:bit

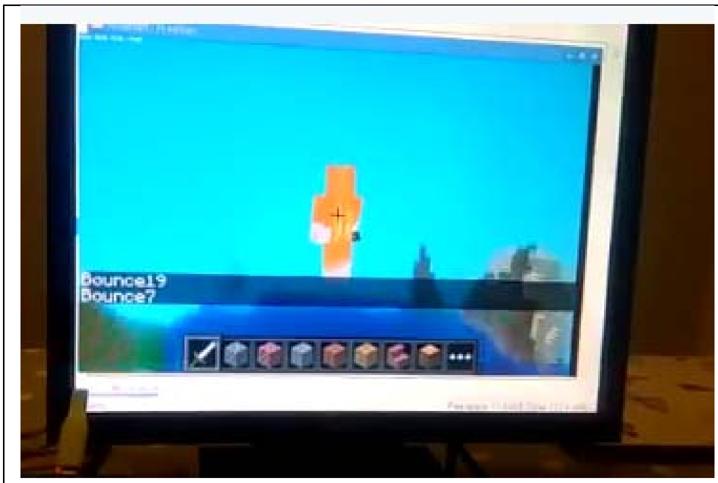
You will need 1 Raspberry Pi, and two microbits for this to work.

```
from microbit import*
import random
import radio

radio.on()

REFRESH = 500

JumpStatus = ["Jump","Donotjump"]

def get_data():
    if button_a.is_pressed():
        a = random.choice(JumpStatus)
        radio.send(random.choice(JumpStatus))  # a-ha
```

```
def run():
    while True:
        sleep(REFRESH)
        get_data()
        # Read any incoming messages.
        incoming = radio.receive()
        if incoming == 'Jump':
            display.scroll("Jump")
            print('Jump')

        elif incoming == 'Donotjump':
            display.scroll("Do not jump")
            print('Donotjump')

#display.show('J')
run()
```

<u>To go on Pi</u>

Picture of teleportation modification



Bounce19
Bounce7

```
"""
Code written by Martin O'Hanlon in the following blog post:
http://www.stuffaboutcode.com/2016/03/microbit-get-data-from-usb.html
"""


import time
import serial
from mcpi.minecraft import Minecraft
from time import sleep
from mcpi import block as block
import random


PORT = "/dev/ttyACM0"
BAUD = 115200


s = serial.Serial(PORT)
s.baudrate = BAUD
s.parity   = serial.PARITY_NONE
s.databits = serial.EIGHTBITS
s.stopbits = serial.STOPBITS_ONE
```

```
#read the first line and flush any bad data

s.readline()


def read_microbit_data():

    #read a line from the microbit,

    data = s.readline()

    data_s = data.rstrip().split(" ")

    JumpStatus = data_s[0]

    print JumpStatus

    return JumpStatus


mc = Minecraft.create()

try:

    playerPos = mc.player.getTilePos()

    while True:

        JumpStatus = read_microbit_data()

        if JumpStatus == "Jump":

            pos = mc.player.getPos()

            #change this to teleport

            mc.player.setPos(random.randint(-100,100), random.randint(1,45),random.randint(-100,100))

            msg = JumpStatus

            mc.postToChat(msg)

finally:

    sleep(1)

    s.close()
```