



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts,Customers Priority,Honest Operation,and Considerate Service",our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

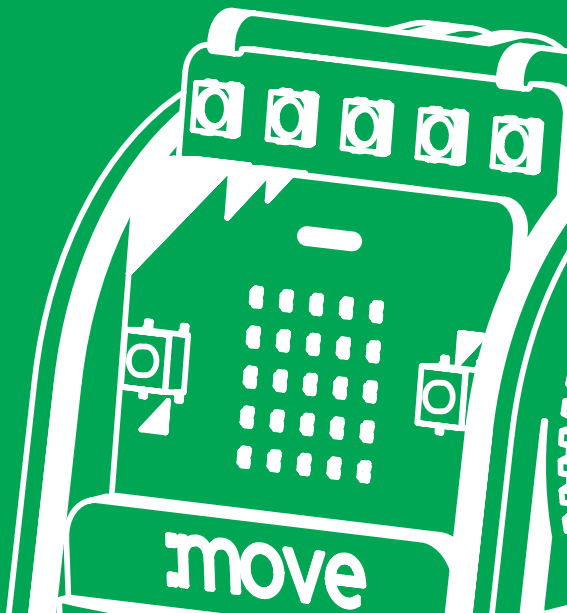


Kitronik

FOR BBC micro:bit
MOVE
mini



V1.0



The **Kitronik :MOVE mini for the BBC micro:bit** provides an introduction to robotics. The :MOVE mini is a 2 wheeled robot, suitable for both remote control and autonomous operation. A range of add-on boards can expand the capabilities to include more advanced functionality.

The included :MOVE servo:lite board can also be used in conjunction with a BBC micro:bit to build other movement based projects.

Visit kitronik.co.uk/movemini for more details.

CONTENTS

INTRODUCTION TO THE :MOVE MINI	2
SUPPLIED WITH THIS KIT	3
GETTING CONNECTED	4
THE BBC micro:bit SOFTWARE	5
PROGRAMMING THE BBC micro:bit	6

ASSEMBLING :MOVE MINI

1. ASSEMBLING THE PCB	7
2. FLASH THE ZIP LEDs	8
3. BUILDING THE WHEELS	13
4. TESTING THE SERVOS	14
5. CALIBRATING THE SERVOS	17
6. ASSEMBLING THE CHASSIS	18
7. ATTACHING ELECTRONICS	22

CODING :MOVE MINI

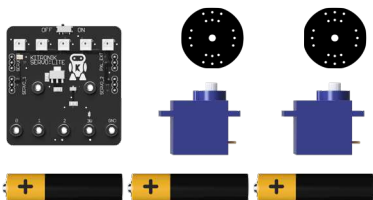
8. CODE MOVE MINI TO MOVE	24
9. DRAW A SHAPE USING JAVASCRIPT	26

SUPPORT FOR :MOVE MINI

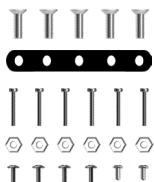
10. GO ONLINE!	30
:MOVE mini PIN OUT	30
TROUBLESHOOTING	31

SUPPLIED WITH THIS KIT

ELECTRONICS



FIXINGS



CHASSIS PANELS

2x

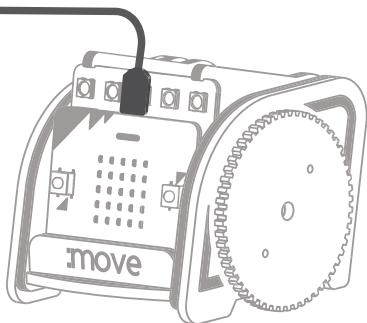


1x

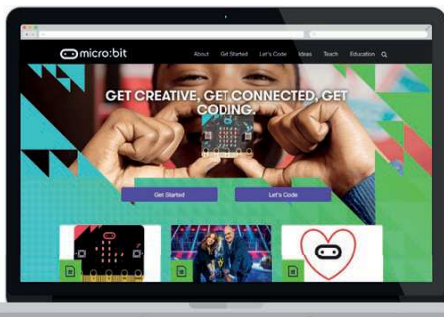


GETTING CONNECTED AND FINDING THE PROGRAMMING ENVIRONMENT

Using a USB to micro-USB type B cable, connect the BBC micro:bit to a computer.



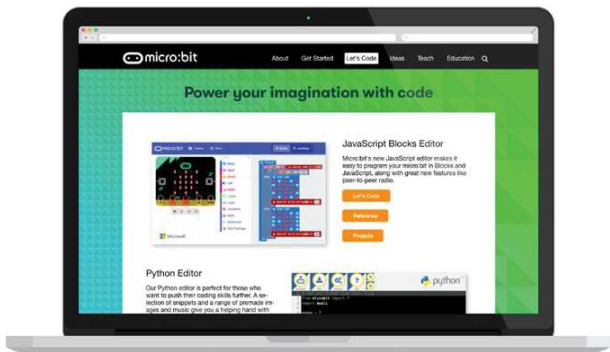
Code will be created on the BBC micro:bit website.
www.microbit.org



THE BBC micro:bit SOFTWARE


The BBC micro:bit is programmed using a web based (internet access required) programming environment which is found at www.microbit.org.

To save programs to access at a later date you will need to be logged in (although you can create a program without doing this). If this is the first time you have used your BBC micro:bit then please refer to our getting started guide at www.kitronik.co.uk/microbit.



All of the experiments in this guide are based on the **Microsoft PXT Block** and **Microsoft PXT JavaScript** editors. The Microsoft PXT Block Editor is a very easy to use graphical editor. The Microsoft PXT Javascript editor is a text based programming language which is ideal for slightly more complex programs. It is possible to convert a Block program into a JavaScript program. This offers an easy way to of progressing from Block programming to JavaScript.

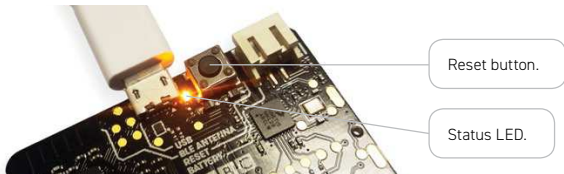
GETTING A PROGRAM ON TO THE BBC MICRO:BIT

It is very easy to transfer a finished program to the BBC micro:bit. First of all select 'download' . This is where the program is converted into a program the micro:bit can understand. This is known as a '.hex' file. If it has compiled successfully, it will return the message 'Download completed... Move the .hex file to the MICROBIT drive to transfer the code to your micro:bit.' The message 'Do you want to open or save microbit-script.hex from microbit.org' will appear. Select 'Save As' from the 'Save' drop down menu and save the hex file to a folder for BBC micro:bit .hex files.

Next plug a BBC micro:bit into the computer via USB. The BBC micro:bit will appear as a removable drive on the computer called 'MICROBIT'.

To download the .hex file to the BBC micro:bit 'Drag' the .hex file from the folder where it was saved and 'Drop' it onto the MICROBIT removable drive. A message will appear saying ' Copying 1 item..... to MICROBIT'. At the same time, the yellow LED on the back of the BBC micro:bit will flash.

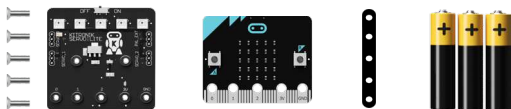
After a few seconds, the download will complete and the BBC micro:bit should now be running the program.



If it doesn't you may need to press the reset button, which is next to the status LED.

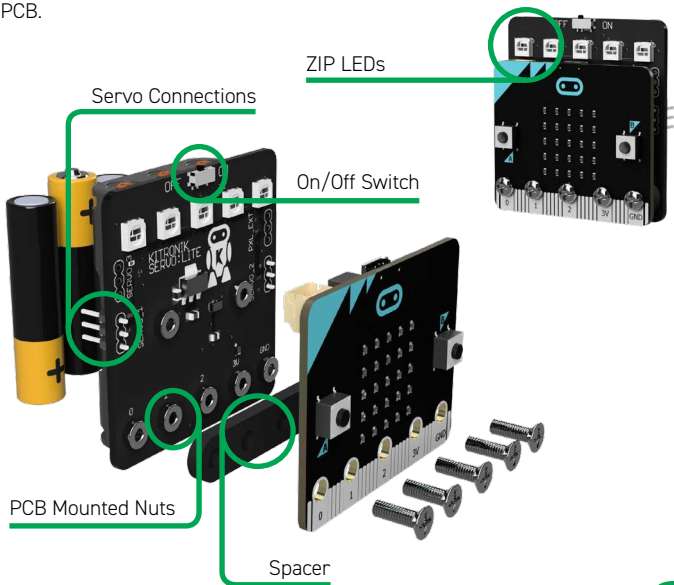
1

ASSEMBLING THE SERVO:LITE PCB



Micro:bit sold separately

STEP 1: Use a small Phillips screwdriver to screw the five M3 machine screws through the micro:bit and spacer, into the nuts mounted on the PCB.



2

FLASH THE ZIP LEDs

STEP 1: Put batteries into the SERVO:LITE PCB, and turn it on.

STEP 2: Connect it to a computer using a micro-USB cable.

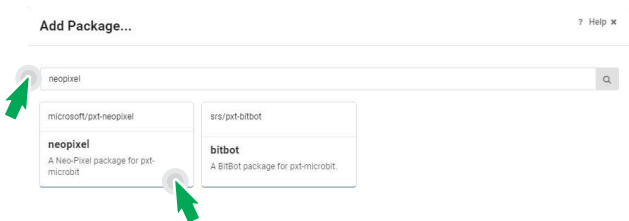
STEP 3: Bring up Javascript Blocks Editor (makecode.microbit.org).

NOTE: Kitronik's ZIP LEDs are compatible with Adafruit's Neopixels.

STEP 4: In the toolbox towards the left of the screen, select the 'Advanced' section. Additional packages should appear below.

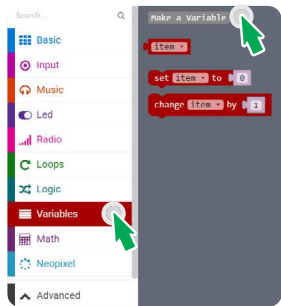
STEP 5: Select 'Add Package'.

STEP 6: In the search bar type 'neopixel', then select the 'neopixel' box.



NOTE: This will load a set of blocks compatible with Kitronik's ZIP LEDs, which makes them really easy to code!

STEP 7: Create a variable and name it 'Pixel Array'.



New variable name:

Pixel Array



WHAT THIS MEANS

A variable is like a container which can store information. This could be a number, a word or a piece of information you want your program to remember.

STEP 8: Create the following code.



WHAT THIS MEANS

This tells the micro:bit that Pin 0 is connected to 5, colour addressable LEDs.

```
on button A pressed
  Pixel Array show color red
```



WHAT THIS MEANS

When button A is pressed, light up all the pixels red.

```
on button B pressed
  Pixel Array clear
  Pixel Array show
```

WHAT THIS MEANS

When button B is pressed, clear the LED's represented by the variable 'Pixel Array'. This will turn them off.



STEP 9: Download!

The program will automatically run on the simulator. Click on the 'A' button on the simulator to see the LED pattern. Click on 'B' button to clear the pattern.



Download the program to generate the .hex file.
Save the .hex file ready to upload to the BBC micro:bit.

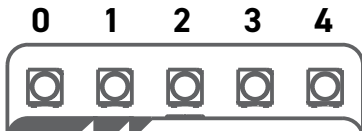


Plug the BBC micro:bit into a USB port then drag and drop the .hex file onto the BBC micro:bit window to upload the program.

STEP 10: Press A and get ready for the lights!

STEP 11: Try changing the code to make a different colour.

As well as addressing all the pixels at once it is possible to set them individually, or as a group. The first pixel always has an address of 0 (see diagram below).



STEP 12: Change the code under 'On button A pressed' to the following.

NOTE: You may need to click on 'More' under the 'neopixel' toolset to view extra blocks.

```
on button A pressed
  Pixel Array range from 0 with 2 leds show color red
  Pixel Array set pixel color at 2 to white
  Pixel Array range from 3 with 2 leds show color blue
```

WHAT THIS MEANS

Set the first two pixels to red, the middle pixel to white and the last two to blue.



REMEMBER: To show a change you must use a block with 'show' in it.

STEP 13:



Download



Upload

STEP 14: Press A and watch the lights!

STEP 15: Create the code below.

```
on start
  set PixelArray to NeoPixel at pin P0 with 5 leds as RGB (GRB format)

on button A pressed
  PixelArray set pixel color at 0 to red
  PixelArray set pixel color at 1 to yellow
  PixelArray set pixel color at 2 to green
  PixelArray set pixel color at 3 to blue
  PixelArray set pixel color at 4 to purple
  PixelArray show

on button B pressed
  PixelArray clear
  PixelArray show

forever
  pause (ms) 100
  PixelArray rotate pixels by 1
  PixelArray show
```



WHAT THIS MEANS

This code shows a colour changing pattern when button A is pressed and stops when button B is pressed.

NOTE: The 'rotate pixels' tool shifts each LED colour onto the next LED. When it reaches the end of the line, it goes back onto to the first LED.

STEP 16:



Download



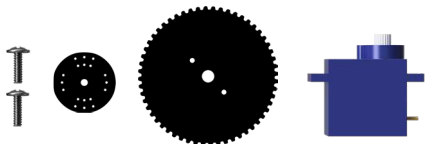
Upload

STEP 17: Press A and watch the light show!

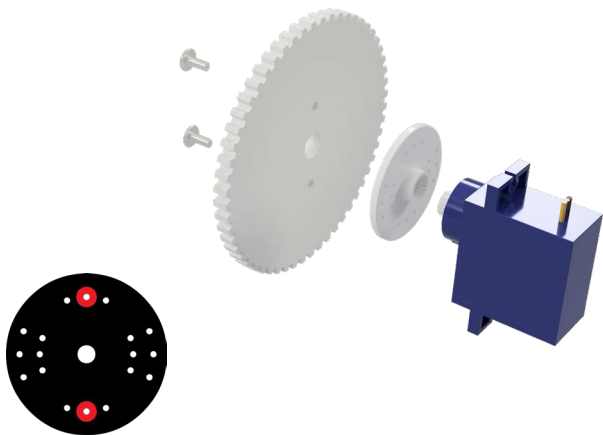
3

BUILDING THE WHEELS

2x



STEP 1: Screw the 2 larger screws from the servo packet, through the wheel and into the appropriate holes in the servo attachment. The centre of the horn and the centre of the wheel should be aligned. Press fit the wheel onto the servo and repeat for the opposite side.



NOTE: The screws should go through the holes shown above in red.

4

TESTING THE SERVOS



SERVOS

The continuous rotation servos used in the :MOVE mini are controlled in the same manner as normal remote control servos. These servos are controlled by a repeating pulse, whose width commands the servo to turn to a position. For a normal servo, position is measured from the output shaft and used to determine what angle the servo should stop at.

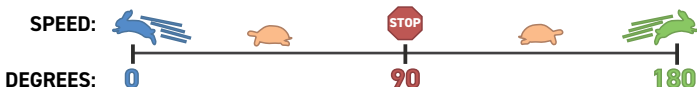
NORMAL SERVO



— 0 DEGREES
 — 90 DEGREES
 — 180 DEGREES

CONTINUOUS ROTATION

A continuous rotation servo is slightly different. Instead of the signal telling the servo how **far** to move, it tells the servo how **fast** to move.



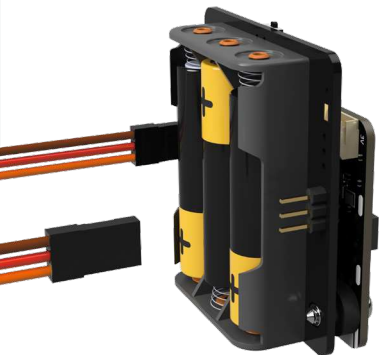
Because of component tolerances, we may need to set the centre point on the :MOVE mini servos to ensure it will stop when commanded. This is done with a trimmer, which is explained later on.

STEP 1: Plug the servos into the SERVO:LITE board.

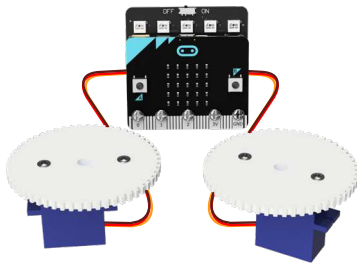
The :MOVE board connections are:



TOP : GROUND : BROWN
MIDDLE : POWER : RED
BOTTOM : SIGNAL : ORANGE



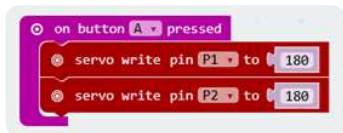
STEP 2: With both wheel servos plugged in, it is time to write some test code. Set out the servos like below. This will allow trimming/calibration of the servos. This means they will stop and travel at the same speed when commanded.



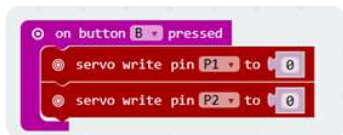
STEP 3: Bring up JavaScript Blocks Editor (pxt.microbit.org).

STEP 4: Create the following code.

When button A is pressed, both servos should turn anti-clockwise (looking from the wheel side).



When button B is pressed both servos should turn clockwise (looking from the wheel side).



When buttons A + B are pressed the servos should stop turning.



If they do not then the centre point trimmer will need adjustment. On the bottom of the servo, there is a small hole. This is used to access the trimmer.

STEP 5:



Download



Upload

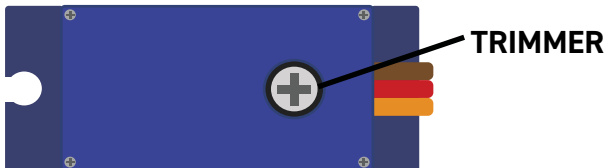
STEP 6: Test out the buttons.

5

CALIBRATING THE SERVOS



STEP 1: Press buttons A+B,
Then with a small screwdriver (through the hole) gently move the centre point trimmer until the servo completely stops. There should also be no sound coming from the servos.



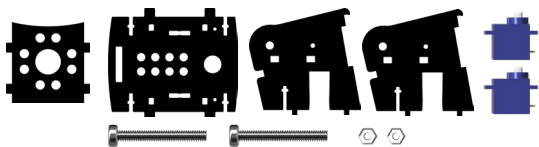
STEP 2: Once the servos are calibrated unplug them from the board and detach the wheels from the servos.

NOTE: The diagram below shows how the number of degrees set in the code relates to the speed of the servo.



6

ASSEMBLING THE CHASSIS

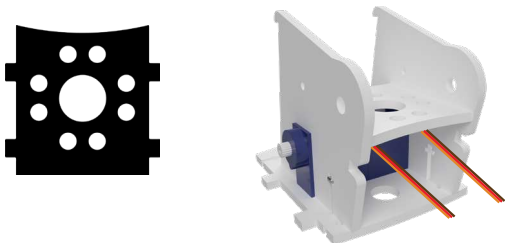


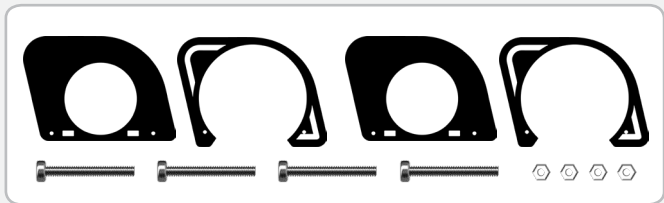
STEP 1: Slot the servo into the base plate with the shaft facing out, and the wires facing up. This should fit very tightly and may require some pressure to snap into place.



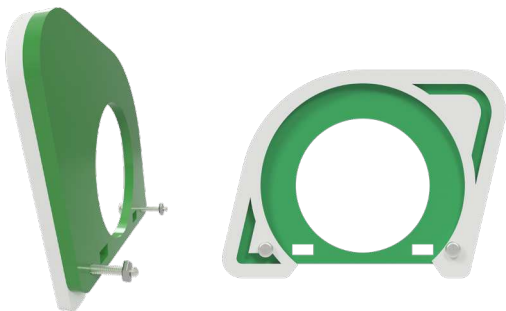
STEP 2: Slot the side part over the top of the servo, ensuring the panel sits in front of blue plastic lip of the servo before securing it with a screw and nut. Then repeat for the opposite side.

STEP 3: Snap the pen mounting plate in between the two vertical plates, just above the servo. The servo cables should pass out of the rear of the chassis.

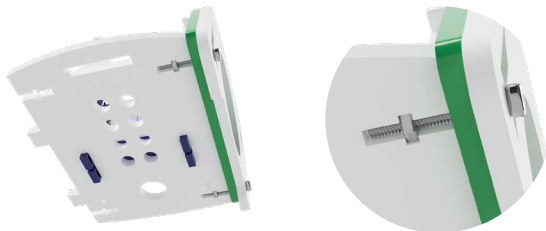




STEP 4: Slot the screws through the outer panel and into the green panel, then add a nut onto the end of each. Repeat for the opposite side.



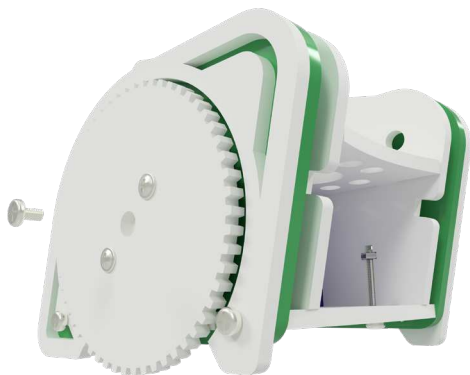
STEP 5: Attach to the chassis, slot the nut into the T-joint and tighten the screw.



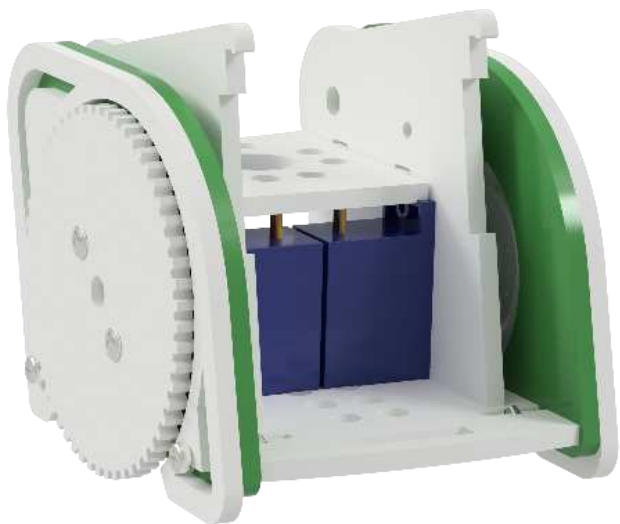
STEP 6: Repeat for the opposite side.



STEP 7: Add a wheel (built earlier) and screw the smallest servo screw through the middle to secure it.

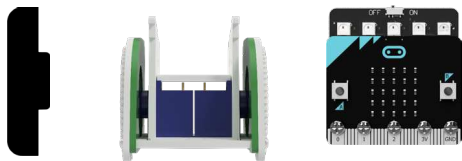


STEP 8: Repeat for the opposite side. The chassis is now complete.



7

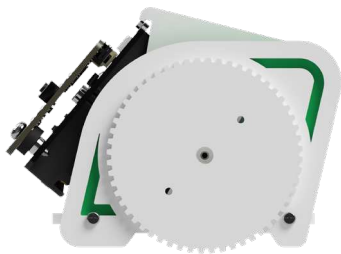
ATTACHING ELECTRONICS



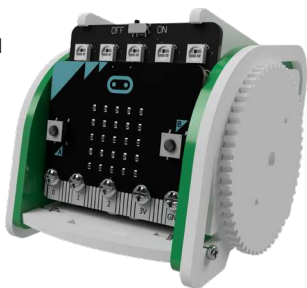
STEP 1: Plug the servo cables back into the SERVO:LITE board. The left servo should plug into the left-hand side of the :MOVE board and vice versa. The cables should be threaded down the gaps on either side as shown and any excess tucked under the pen mounting plate.



STEP 2: Clip the SERVO:LITE board under the hooks on the inner side plates and slide it between the outer plates.



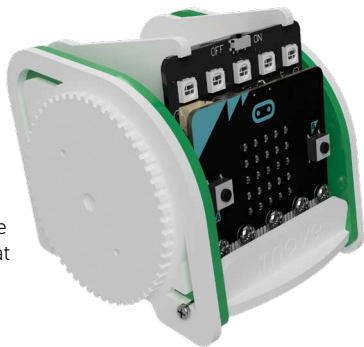
STEP 3: Push the SERVO:LITE board fully back inside the :MOVE mini.



STEP 4: Secure the SERVO:LITE board and micro:bit with the :MOVE T-piece.

STEP 5: Tidy up the cables by tucking them into the gaps in between the side panels.

STEP 6: Excess wires can then be fed into the servo compartment at the rear.

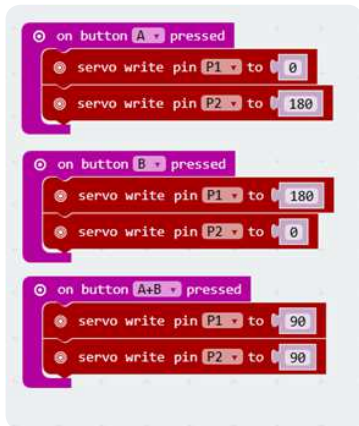


8

CODE :MOVE MINI TO MOVE

DRIVE IN A STRAIGHT LINE

STEP 1: Create this code.



NOTE: If the move mini does not travel in a straight line then you can adjust the values to make the servos run at the same speed. You may also need to adjust the trimmer (see pg 17).



WHAT THIS MEANS

When button A is pressed, drive full speed forwards. When button B is pressed, drive full speed backwards.

NOTE: Because the servos are mounted on opposite sides of the :MOVE mini, one has to turn clockwise (Value 0), and one anti-clockwise (Value 180) to drive the :MOVE mini in a straight line.

STEP 2:



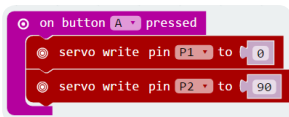
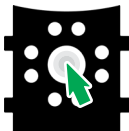
Download



Upload

DRAW A CIRCLE!

STEP 3: Drop a marker through the designated hole on the pen mounting plate and create the code below.



WHAT THIS MEANS

Drive one servo at full speed, and the stop the other. This will cause the :MOVE mini to drive in a circle.

STEP 4:



Download



Upload

ADD SOME LIGHTS!

STEP 5: Create the following code to give :MOVE mini a head-light or tail-light.

