UCL Micro:bit Robotics Documentation

Release 1.0

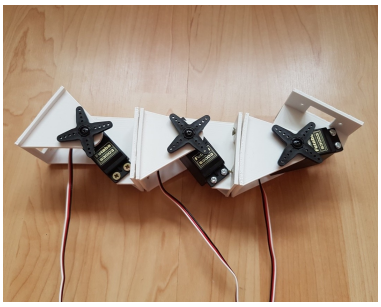
Rae Harbird

May 18, 2021

Contents

This project is designed to give students an introduction to robotics with a robot inspired by the natural world. There's no need to know how to code before diving in but we can assure you that once you have tried it, you won't want to stop. The project was designed by Professor Stephen Hailes, a robotics expert and dedicated educator. You will use the micro:bit to make the robot move. Professor Stephen Hailes has written a servo motor library for the micro:bit to make this easier.

The robot is a caterpillar (or maybe a snake, depending on the motion style) which was designed by Dr Juan González-Gómez.



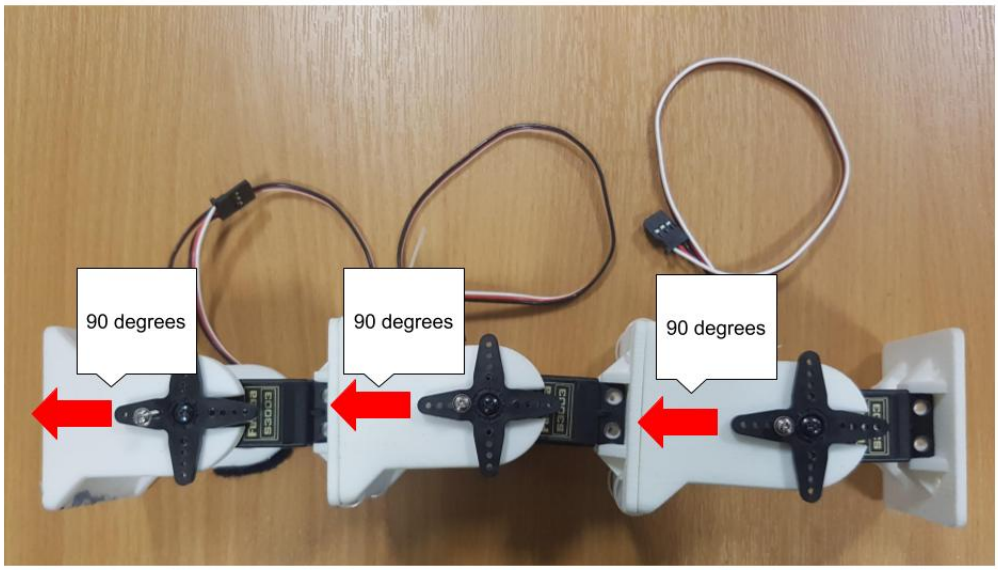
Caterpillar robot designed by Dr Juan González-Gómez

CHAPTER 1

Building Your Own Robots

Take a look at Dr Gómez's instructions for making and building the units for the robot. The latest design is here. You can also use the older design, found here.

Note: When the robot is flat on the table the servo motors should be set to 90°. You will need to set the motors to a 90 degree angle before attaching the units together.



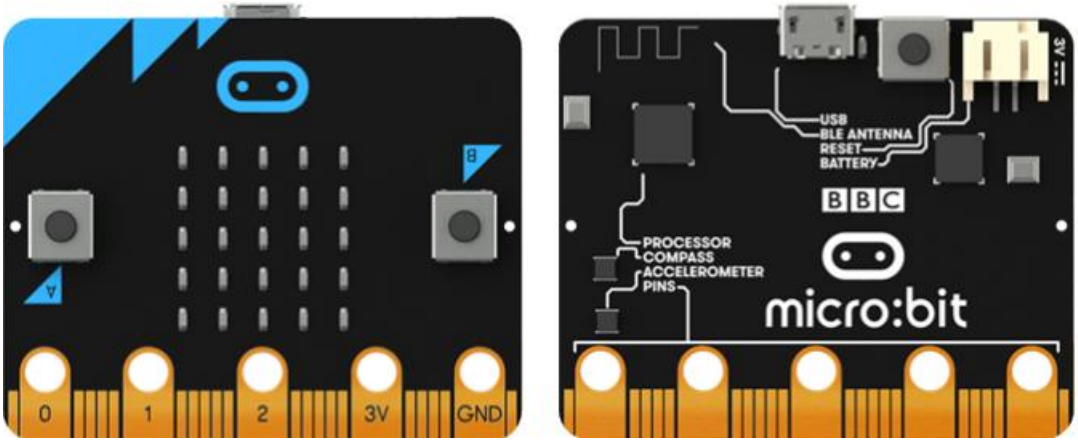
CHAPTER 2

Contents

2.1 Micro:bit - Getting Started

The BBC micro:bit is a tiny computer that you can use to create all kinds of projects from robots to musical instruments – the possibilities are endless. There are a myriad of features that you can use in your designs:

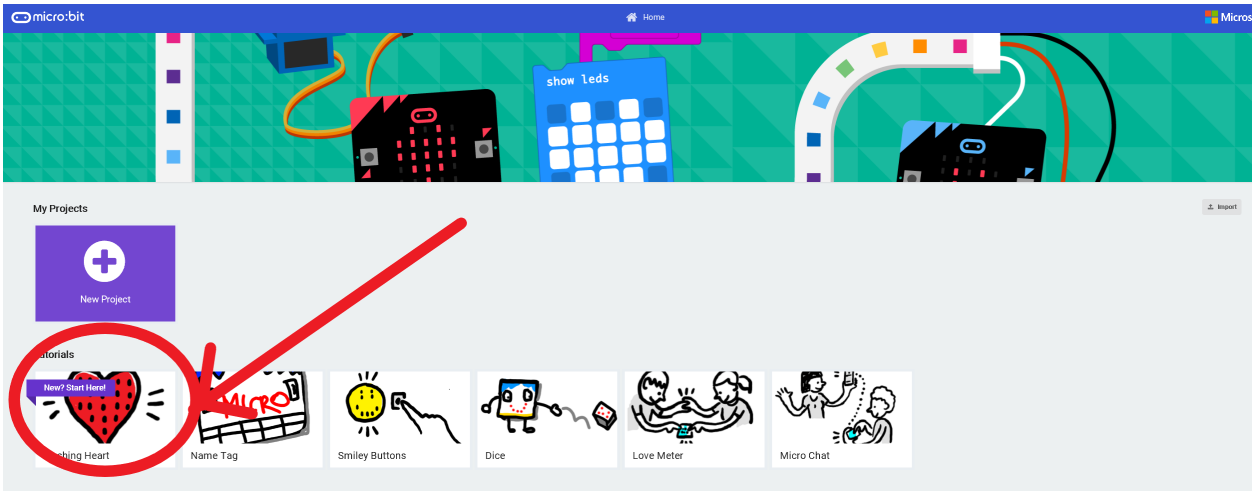
- 25 red LED lights that can flash messages.
- Two programmable buttons (A and B) that can be used to tell the micro:bit when to start and stop things.
- A thermistor to measure the temperature.
- A light sensor to measure the change in light.
- An accelerometer to detect motion.
- A magnetometer to tell you which direction you're heading in.
- A radio and a Bluetooth Low Energy connection to interact with other devices.



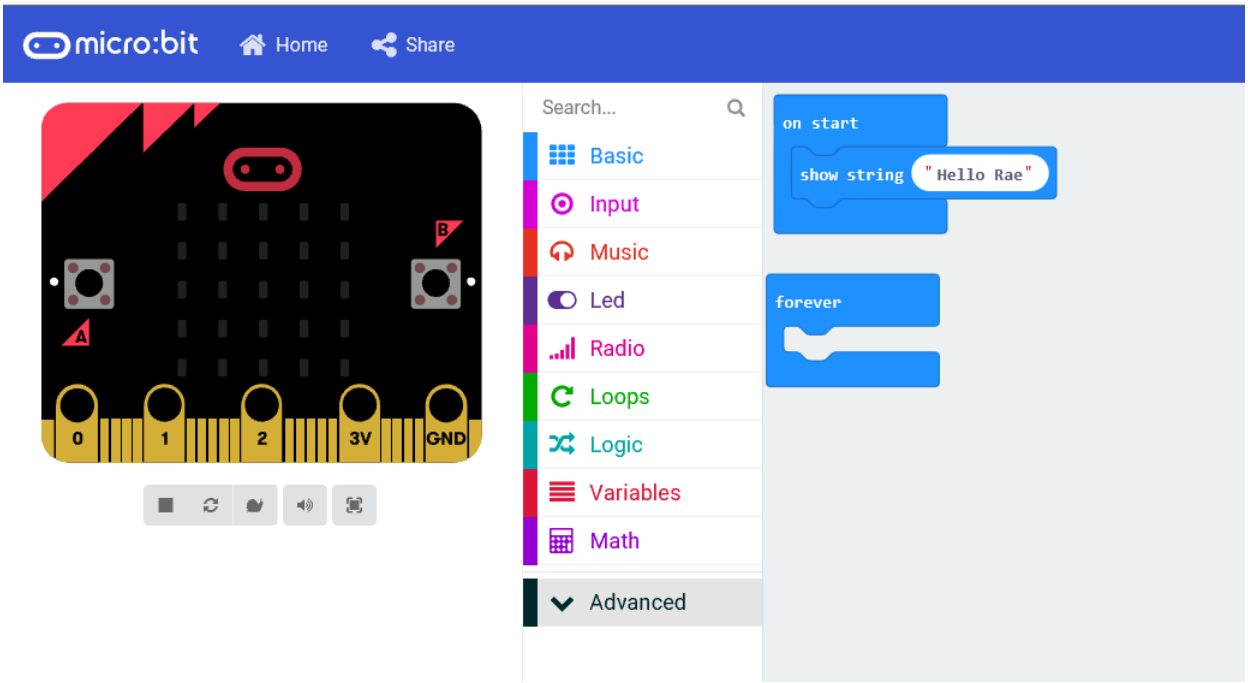
Let's create your first micro:bit program; after that, it's really up to you to discover more.

2.1.1 Your First Program

Go to the micro:bit code editor here: https://makecode.microbit.org, the homepage should look similar to the picture here, click through on and complete the Flashing Heart tutorial:

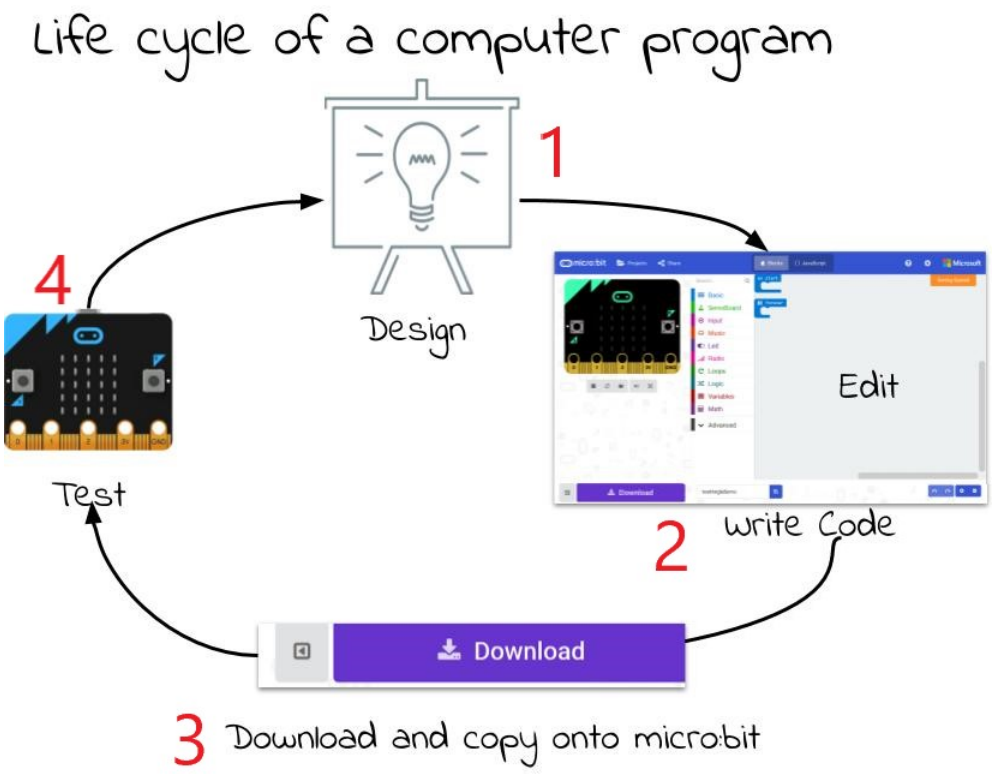


Can you make a change to your code? How about making the micro:bit display: “Hello <Your name>”? When you’ve done that, find out what happens when you put the code in the start block instead of the forever block like this:



Well done, you have written your first programs. Carry on and see what else you can do; try some of the other projects and search for others online.

Coding using the micro:bit is composed of these 4 steps. You can expect to go around the loop quite a few times before you get your code working.



2.2 Crawling Robot

2.2.1 Aims

We're going to build a crawling robot that's inspired by a snake or a caterpillar but how does a crawling animal move? Click on the pictures below to view the video clips:

BBC How Snakes Move

Caterpillar on the move (Lampe, 2013)

Or this one:

Hairy Caterpillar, Caterpillar Moving (HowToDrawCartoons, 2011)

You can see from the videos that a wave passes along the caterpillar so we'll try to make our robot mimic that.

For this project, the robot we are going to build will only have 3 body segments, we're going to need some power to move each segment and we will use a motor, called a servo motor to do this. We'll control the motors using a program written on a micro:bit. In the next section, we'll find out about motors.

2.3 Servo Motors

Let's begin by taking a look at the servo motor. There are two types of servo – the first type is one where you can choose an angle to which the arms should rotate; the second goes round and round at a speed you set. We're going to use the first type shown here:

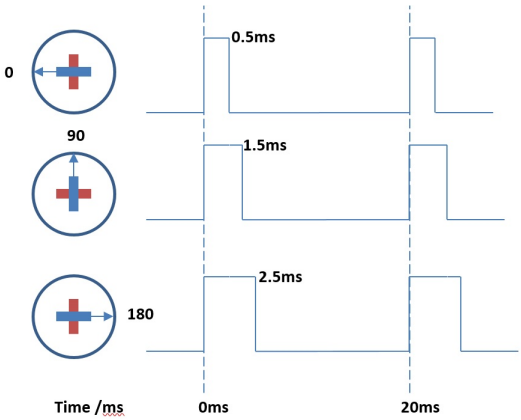
Futaba S3003 servo motor

Servos are used all over the place in both daily life and in robotics. A servo might retract the tray of a DVD player or might be used in radio controlled vehicles and aeroplanes to control the rudder of a boat, or the ailerons - the moveable flaps on the wings of a plane.

An animation of an airplane rolling via its ailerons (NASA, 2013)

2.3.1 Pulse Width Modulation (PWM)

The angle of the arms on the motor is controlled by sending the motor small electrical pulses. This technique is called a pulse width modulation (PWM). Look at the diagram below: a pulse of 0.5 milliseconds (ms) will cause the motor to set the rotor to 0°. A longer pulse of 1.5 ms will set the rotor to 90° and a pulse of 2.5 ms will move the rotor to 180°.



The length of the electrical pulse determines the rotor position.

In the next section, we'll learn how to connect the motors to the micro:bit.

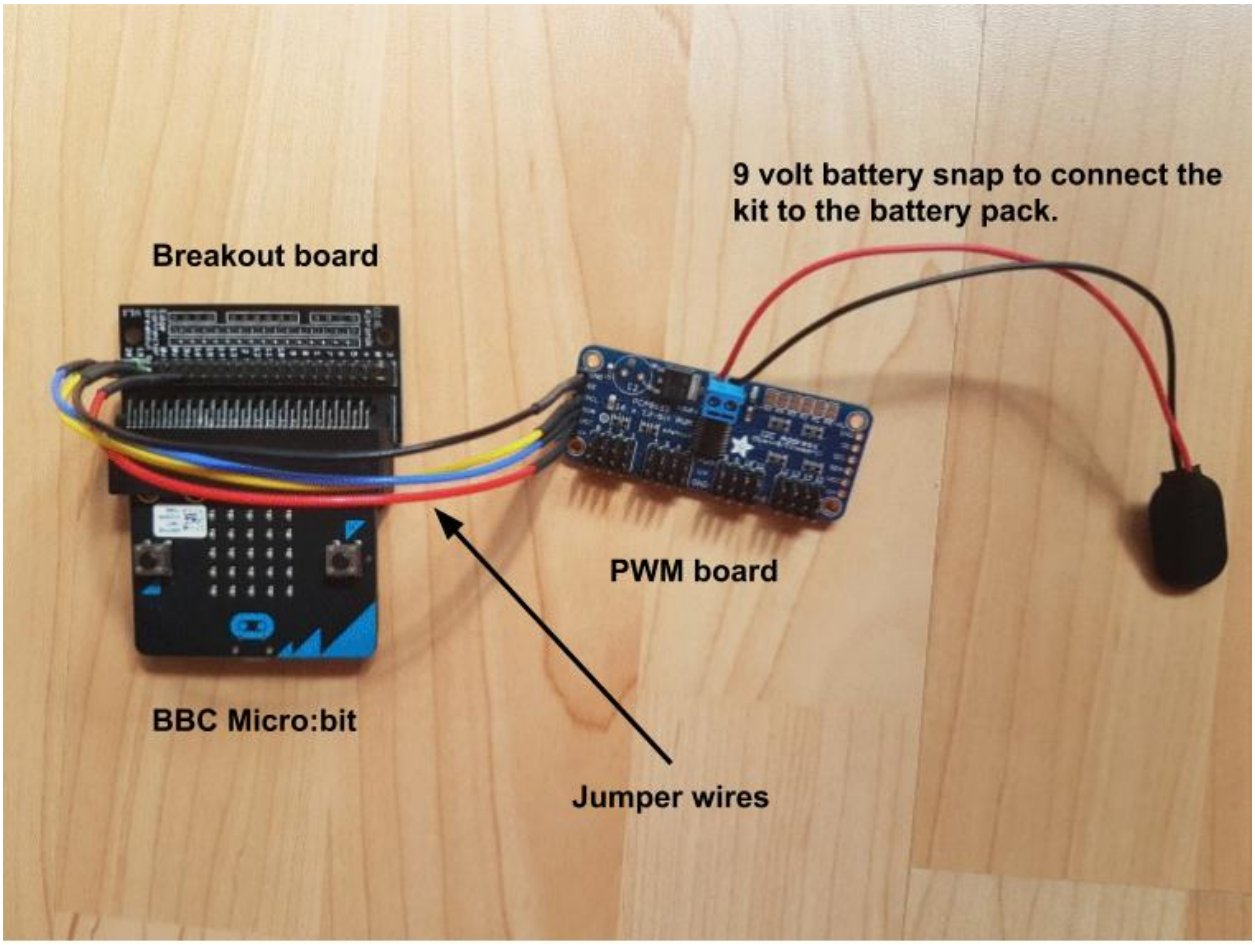
If you want to learn more about using motors with the micro:bit you can watch this video clip.

2.4 Connecting the Parts

The robot is made from parts designed by Dr Gonzales-Gómez. Instructions for the latest design is here. You can also use the older design, here.

Although the micro:bit can drive up to 3 small servo motors, we have decided to use a second board, a PWM board, to connect the micro:bit to the servos. We've done this for two reasons:

- The micro:bit works on a 3.3V circuit but the servos we use work on a circuit of about 5V. This means that we can't just connect the two together anyway.
- We can connect up to 16 servos to the micro:bit using an interface board so we could make a very long snake or caterpillar.



High level view of the micro:bit, breakout board and PWM board

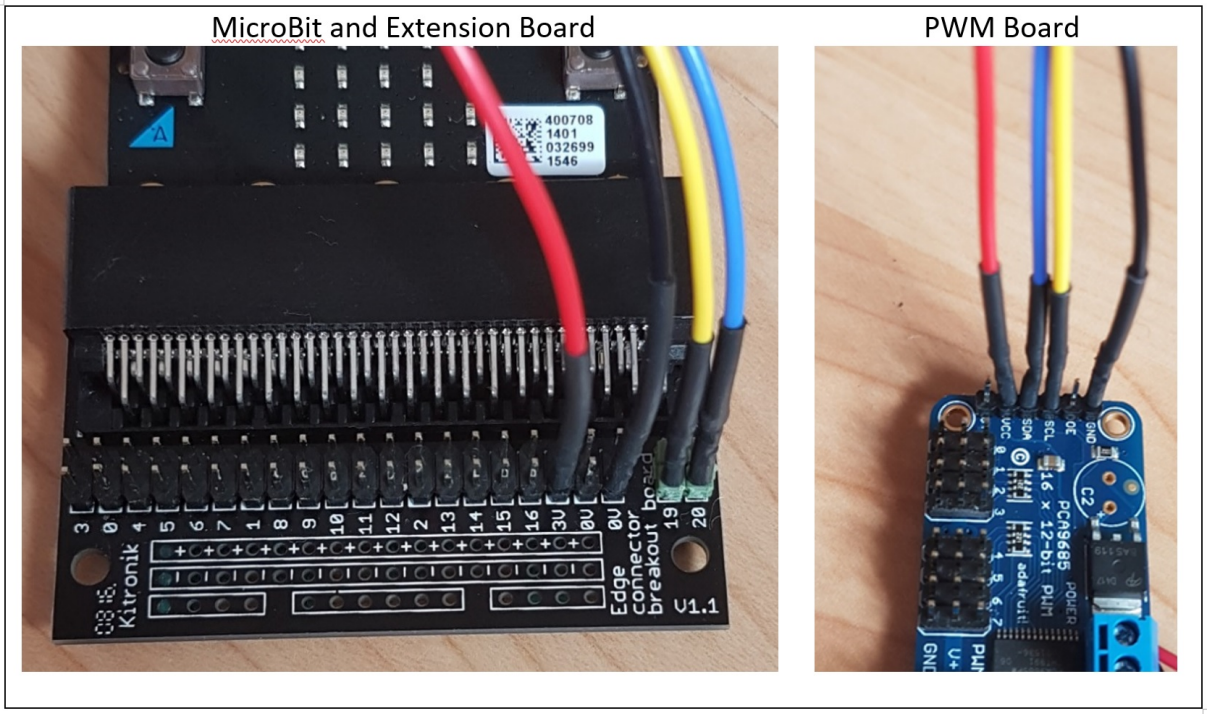
In this picture you can see that the micro:bit is plugged in to a breakout board which is connected, in turn, to a PWM board using some small wires. The PWM board communicates with the micro:bit using a special digital communications channel called I2C. We have written some code to hide these details from you so that you can focus on getting the caterpillar moving.

2.4.1 How to connect the parts

Follow the steps below to make the connections but please note:

DO NOT CONNECT THE BATTERY TO YOUR KIT UNTIL YOUR CIRCUIT HAS BEEN CHECKED

Step 1: Connect the micro:bit to the PWM board

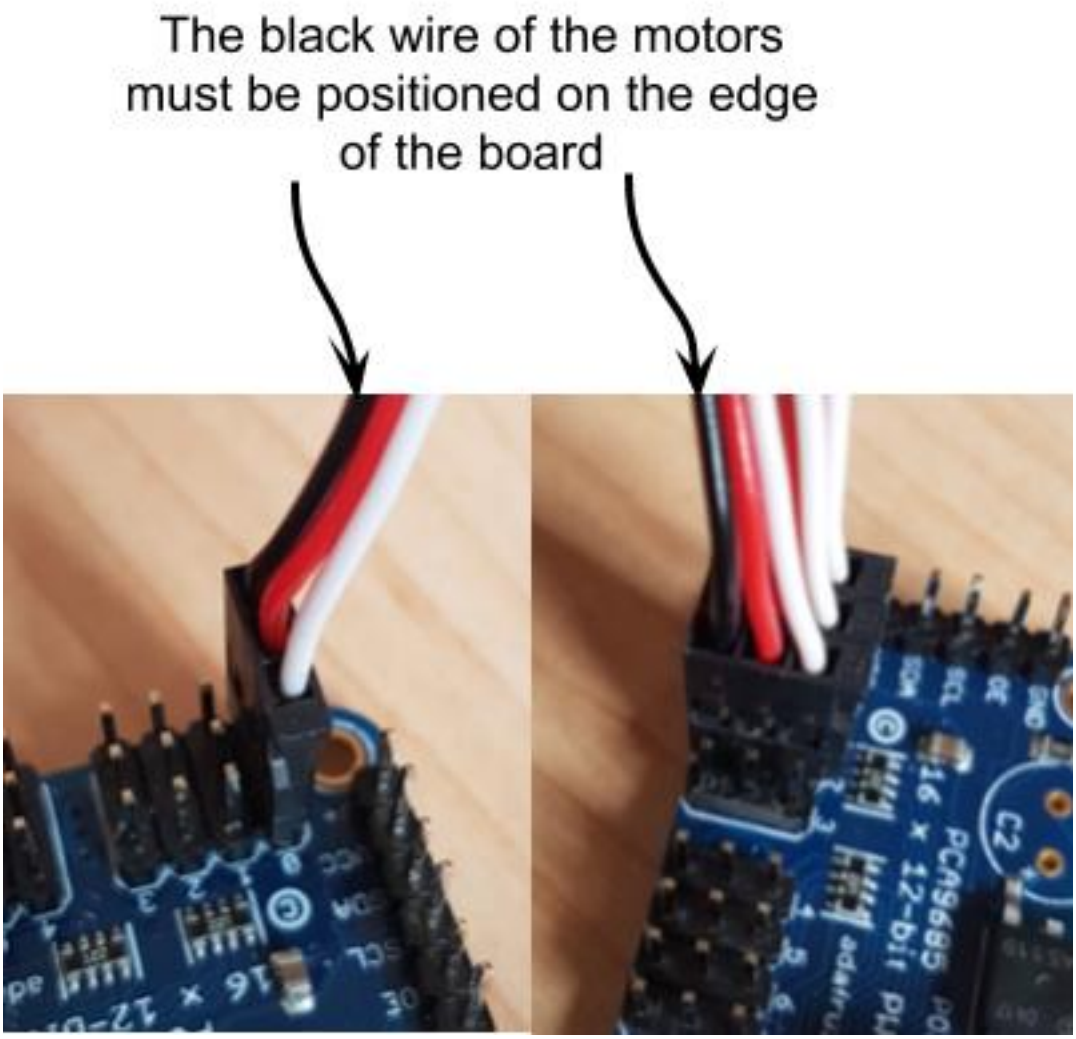


Connecting the micro:bit to the PWM board

You will need 4 female to female jumper wires and you must connect the right pins together. Use this table to match up the pins.

Connection Table		
Purpose of the pin	Micro:bit pin label	PWM board pin label
Power	3V	VCC
Ground	0V	GND
Clock Line	19	SCL
Data Line	20	SDA

Step 2: Plug the motors into the PWM board



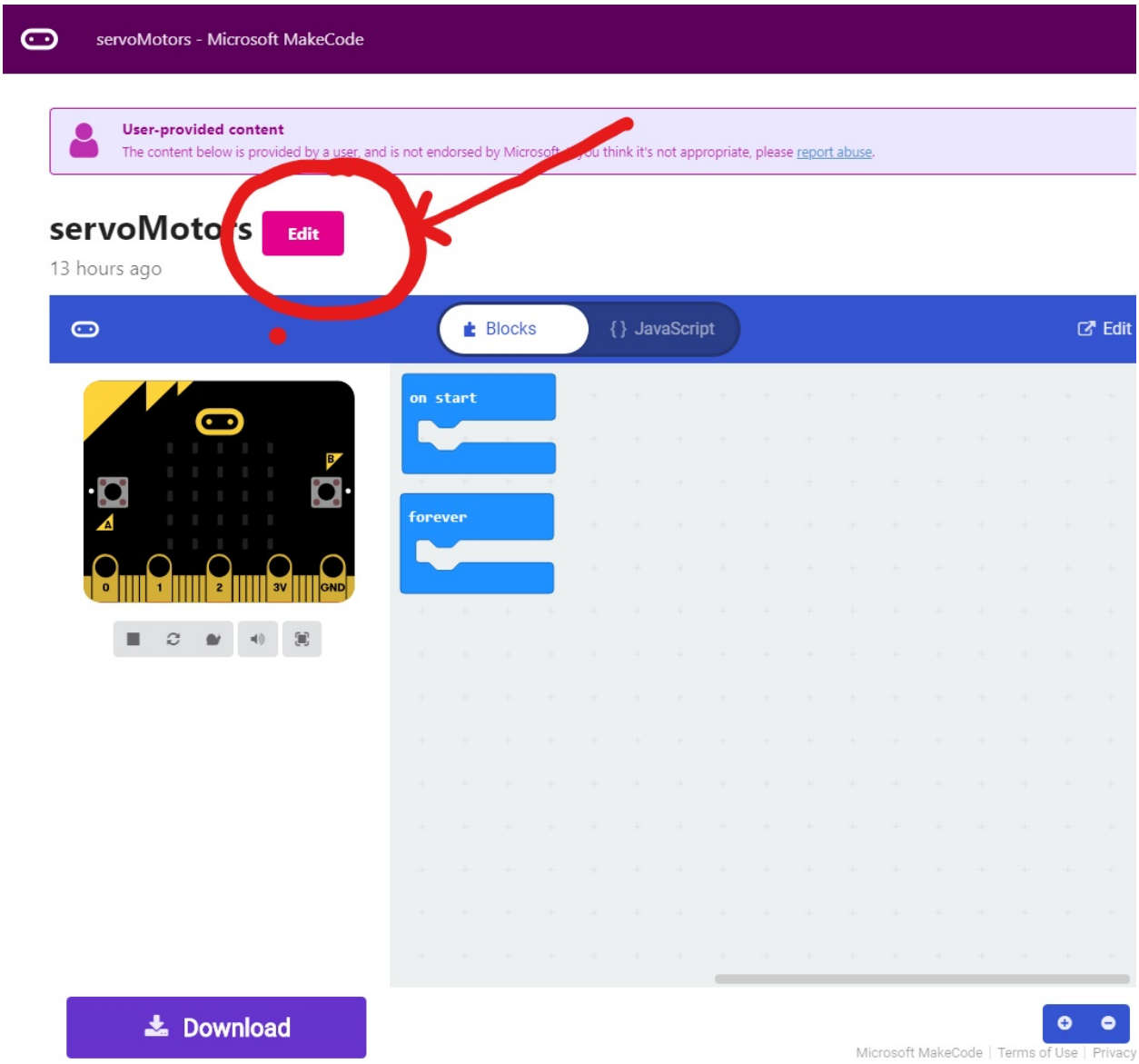
Plugging the motors into the PWM board

The servo motors should be connected to the PWM board this way round with the black wire on the outside of the board. You can see this clearly on the left where there is a servo connected in position 0. It's easier if you start at 0 and work up. You can start off by just connecting 1 motor and add more as you use them.

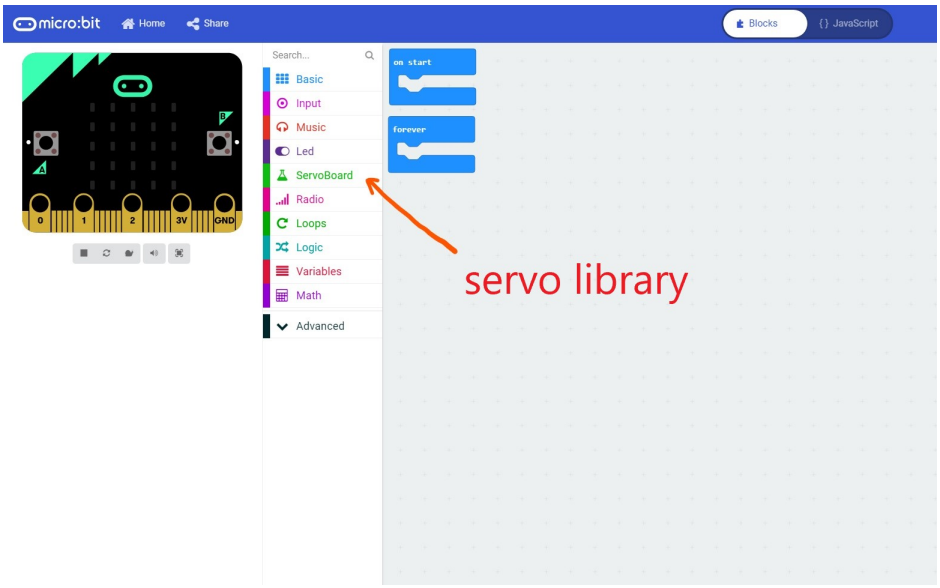
2.5 Get ready to code

We will be programming the micro:bit using the makecode programming environment with some added software to drive the motors. Open a browser, and go to this URL: http://tiny.cc/mbit-servos

When you see the screen below, press the `Edit` button.



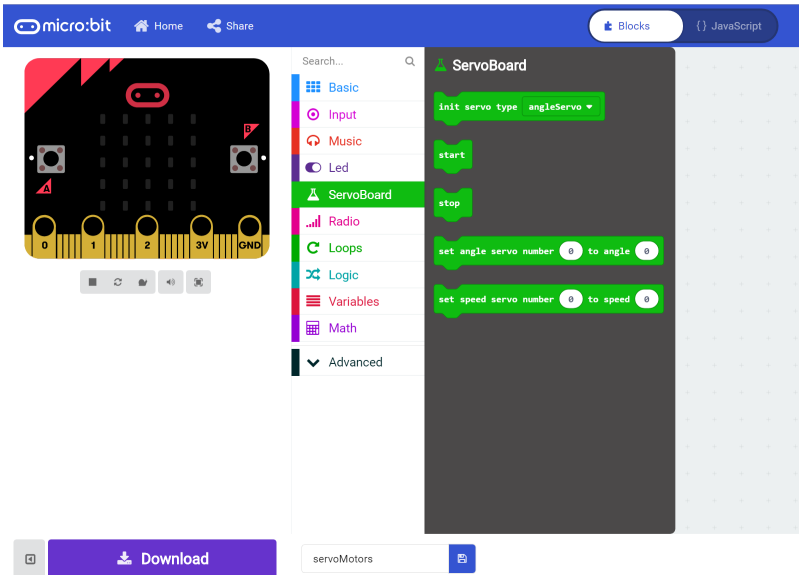
You should see a screen like the one below and you are now ready to make the micro:bit, and your robot, do something.



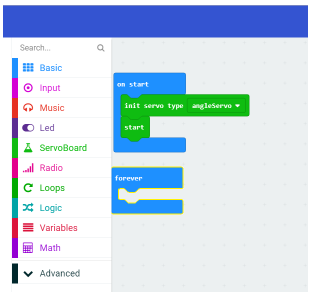
2.6 Moving motors

2.6.1 Set up the motors

Click on the `ServoBoard` menu so that you can see the blocks.

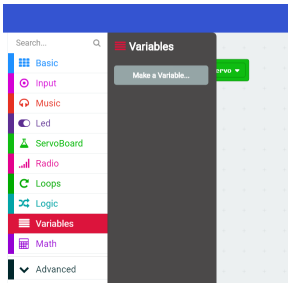


Now drag out the blocks that will set up the motors and place them in the `on start` block so that your code looks like this:

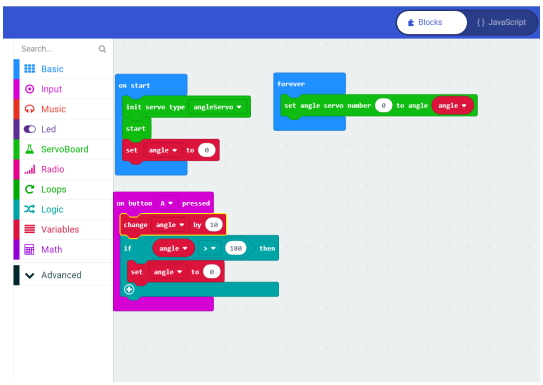


2.6.2 Make the motors move

Now we will add some code to move the motor every time that you press the button. First of all we will need to keep a counter, called a variable, in which we will store the current angle of the motor. You can do that by opening the `Variables` menu and choosing `Make a Variable`, name the variable `angle`.



Now you can add the rest of the code to change the angle of the motor when the button is pressed.



What happens when the angle reaches 180°? Try it out.

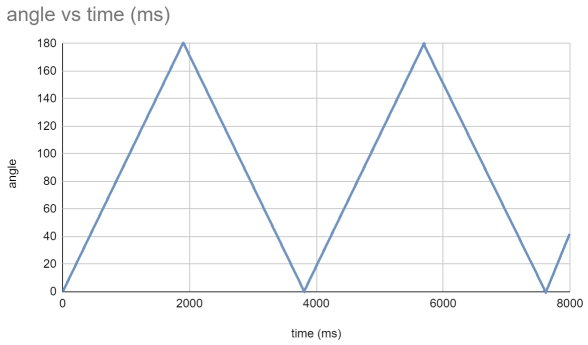
2.7 Forward and Back

Well done, you have got the motor moving and you can see that it moves from 0° to 180° as you press the button. Now we need to change the code so that the motor moves without human intervention so that our caterpillar can crawl.

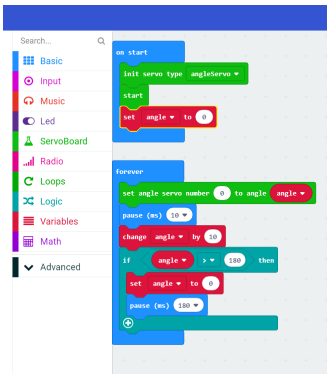
2.7.1 New code

We are going to move some of the code around. Any code that you put in the `forever` loop will keep repeating over and over as its name suggests. Change your code as follows:

1. Move the code in the `on button A pressed` loop to the `forever` loop.
2. Delete the `on button A pressed` loop.
3. Take a look at the diagram below, you will see that the motor takes almost 2000 milliseconds to move through 180°. Add a pause after the `angle` variable has been reset to 0. This is so that the micro:bit will wait for the motor to move to angle 0° before moving it again.
4. Add another pause so that the micro:bit will wait for the servo motor to move each time around the `forever` loop.
5. Experiment with the duration of the pauses until you have a smooth motor motion.



Your code should now look like this:



Experiment

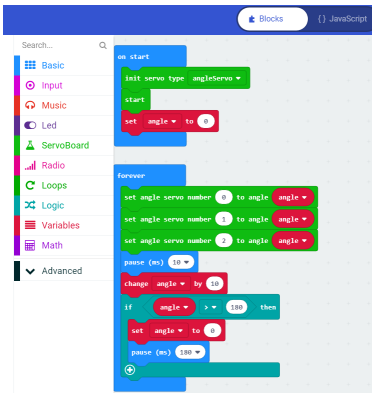
Try the program out. Upload your code to the micro:bit. Did it work? Keep experimenting until you are satisfied with the outcome. Remember that when your robot is lying flat on the table, the motors are set to 90°.

2.8 More than one motor

You are now ready to move more than one motor. Make sure that you have all the remaining motors plugged in to the PWM board and check that you have connected them in the correct orientation.

2.8.1 Servos 1 and 2

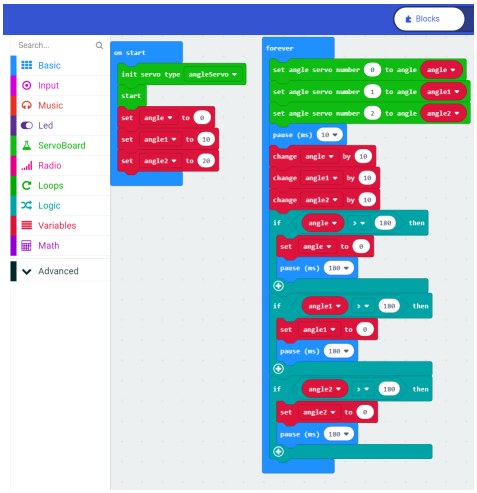
So far your code just moves the motor plugged into slot 0 on the PWM board. You should add code to move the other two motors as well, like this:



You will notice that all of the motors are moving together at the same angle. This is progress but the caterpillar is not going to move very far!

2.8.2 Is it crawling?

Now it's time for you to experiment. Make two more variables `angle1` and `angle2` and initialise the new variables in the `start` loop. Add more code to increase the value of `angle1` and `angle2` in the `forever` loop and don't forget to check whether the value of the angle is greater than 180°.

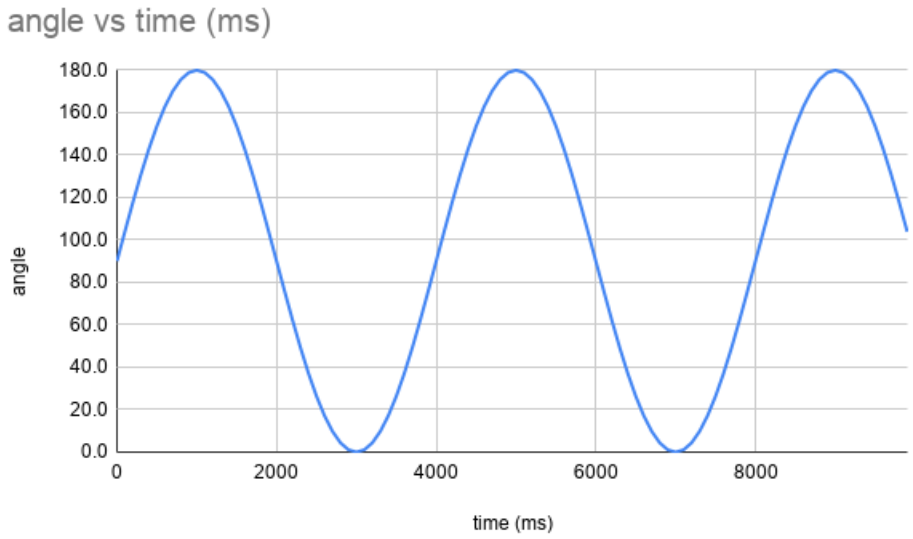


Try it out. Upload your code to the micro:bit. Did it work? Keep experimenting until you are satisfied with the way that the caterpillar crawls.

2.9 Doing it as a sine wave

This is an extension activity.

As it happens, many things in biology happen like sine waves. That's true for this kind of movement - what we'd like to see is the following:



What this means is the change in angle is slower the closer you get to 180° and to 0°. So it starts slowly, then gets faster, then slows down, changes direction, and so on.

The function for the angle is: `angle = 90 + 90 * Math.sin(2 * PI * running_time() / 4000)`

Note: the `Math.sin()` function requires that the number of degrees is expressed as radians. 360° is 2 * PI radians.

- `running_time()` is the amount of time since the micro:bit was turned on in ms.
- The 4000ms is the amount of time it takes to go from angle 0° back to 0° again. Make this number bigger to go slower.
- Sine values vary between 0 and 1 so the angle varies between 0° and 180° as things stand. The `90 *` is the amplitude of the wave and here we want it to go through all 180° of motion. Make this smaller if you want less; if the value was, say 40, then the movement would vary between 60° and 130°.
- The `90 +` is the centre position position of the movement. The servo moves equally either side of that centre. You could choose to centre the movement in a different place. If this was say, 100 and the amplitude was 40 then the servo would move between 60° and 140°.
- You could create variables for amplitude and time period, say `A` and `T`, with another for the centre `C`. This makes the formula: `angle = C + A * Math.sin(2 * PI running_time() / T)`

Congratulations! You made an oscillator. The nice thing about sinusoidal oscillators is that putting things out of phase is easy. Say you wanted two sine waves 90° out of phase then set:`angle1` = sin(t) and `angle2` = sin(t + 90)

2.10 References

Ove Daae Lampe. (2013). Caterpillar on the move. [Online Video]. 2 September 2013. Available from: https://www.youtube.com/watch?v=fRVGWCSij_M. [Accessed: 28 April 2018]

HowToDrawCartoons. (2011). Hairy Caterpillar, Caterpillar moving. [Online Video]. 21 August 2011. Available from: https://www.youtube.com/watch?v=a9Km0edRFG4. [Accessed: 28 April 2018]

NASA (Glenn Research Center, NASA). (2006), [Public domain], via Wikimedia Commons, 10th February 2006. [Accessed: 28 April 2018].

To download this documentation in pdf, epub or html format, click on the link at the bottom of the sidebar on the left.

If you would like to contribute to the documentation or create your own flavour, go ahead! Use git to fork the repository. The documentation is hosted on *ReadTheDocs <https://readthedocs.org>*.