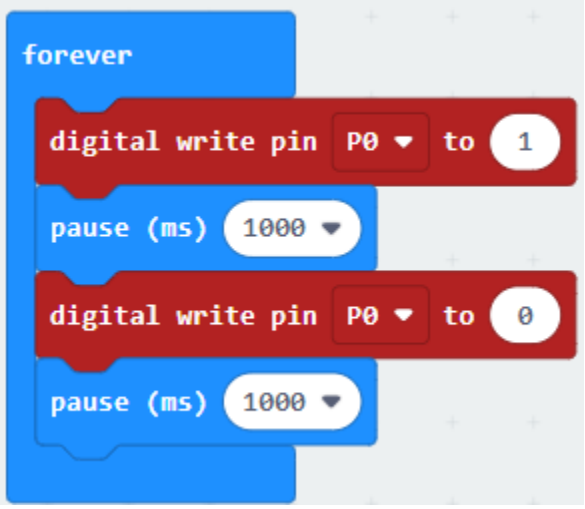
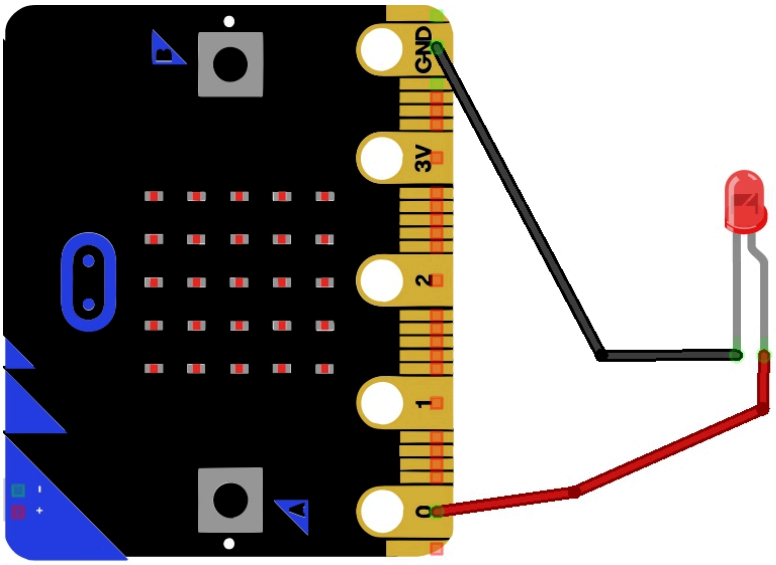
CHALLENGE 1a: Micro:bit, blink 1 LED

1. Connect the ground (GND) port on the micro:bit to the ground(−) lead of the LED
2. Connect port 0 to the positive(+) lead on the LED
3. Write code to turn the LED on (1), then pause for 1 second, then turn the LED off (0) and pause for 1 second.
4. Change the pause time from 1 second (1000 milliseconds) to another value of your choosing. See how that affects the blinking. Experiment.

Things to consider:

- Computers run code very fast, without the pauses, the board will run the on - off code so fast, the human eye won't be able to see so the LED may look off or on even though it is actually blinking.
- You need a line of code (or block) to tell the board to send electricity to the port (also called 'pin') to turn the LED on. To turn the LED off again, to make it blink, you need a line of code that stops sending electricity to the pin.
- In computers, there are two ways to read or write information. One is "digital" the other is "analog".
    - Digital is either ON or OFF, either TRUE or FALSE, either ZERO (0) or ONE (1). So, a value of 1 is on, a value of 0 is off.
    - Analog allows the code to select from a range of values from ZERO (0) to 255.





CHALLENGE 1b: User Input to begin the blinking:

- If you want the LED to blink ONLY for a certain number of times, you could use a Repeat Loop block. This will also need an event of some sort to tell it when to begin. Options for events/user Input:
    - When one of the 2 buttons is pressed, when both buttons are pressed
    - When the touch-sensitive Logo is touched
    - When the board is shaken
    - When the board is tilted in a specific direction

- If the level of sound or light is at a certain level

CHALLENGE 1c: Make the LED blink in different patterns, for instance: on for 1 second, off for half a second, on for 2 seconds, off for 100 milliseconds, on for 3 seconds, off for 3 seconds, etc.. Think about how you could stack a bunch of blocks to make that happen.

Things to consider:

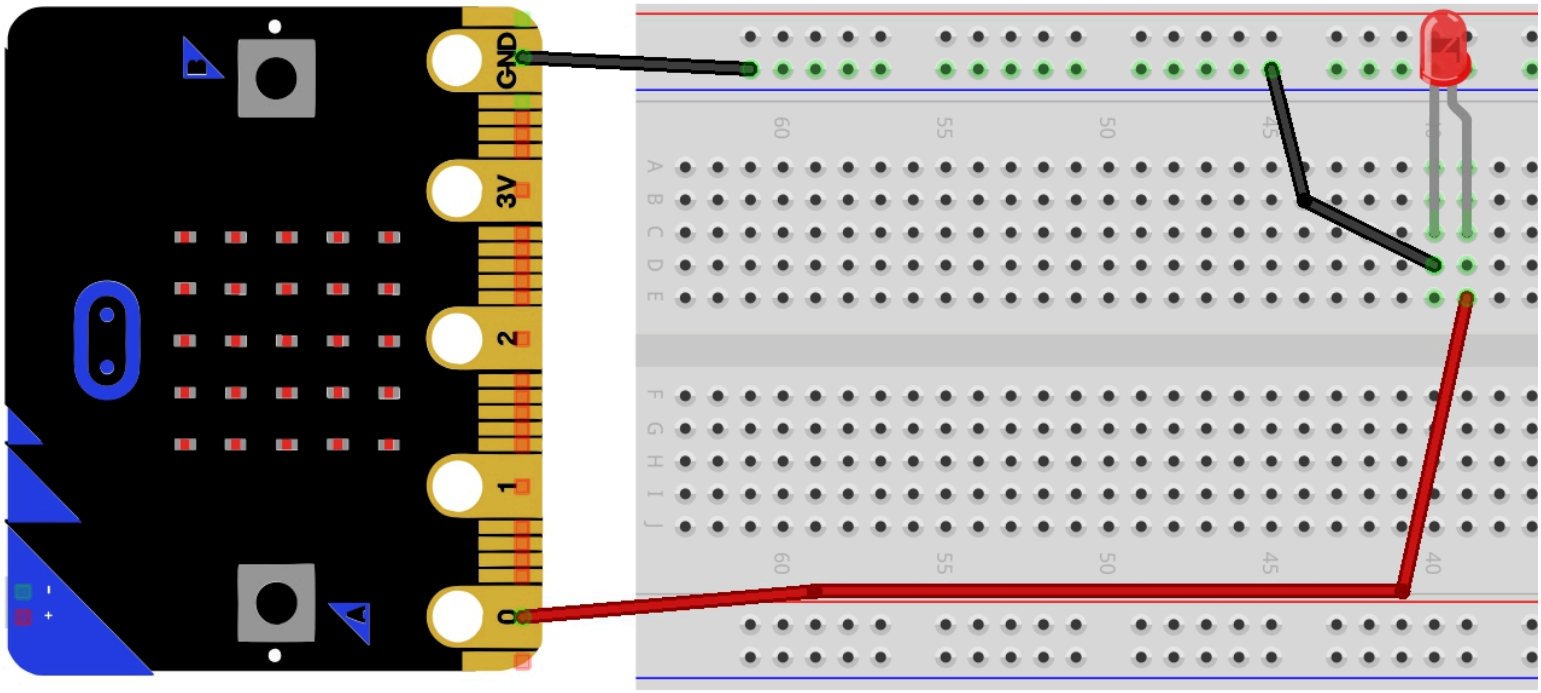
- The current code to blink is only repeating the SAME short, limited pattern. On, pause, off, pause, repeat…
- How could you use additional digital write blocks with pause blocks of different values?

CHALLENGE 2: Using a breadboard and male-alligator clip jumper wires with our blinking LED

1. Use an alligator-to-male jumper to connect the ground (GND) port on the micro:bit to the ground(−) RAIL on the breadboard
2. Place the LED on the breadboard remembering which is the longer (+) positive lead and which is the shorter (−) negative/ground lead
3. Place a jumper in a breadboard hole that is in the SAME ROW as the negative(-) shorter lead of the LED
4. Connect an alligator-to male jumper to port 0 and place the male end in a hole in the SAME ROW as the positive(+) lead on the LED
5. Test your existing LED Blink code to be sure you have wired your breadboard/LED & micro:bit correctly.

Things to consider:

By placing a jumper in the negative rail of the breadboard, I can now ground many LEDs or components, not just one because there is a conductive metal strip that runs along the rail so any wires placed there are connected to the GND port/pin on the micro:bit.

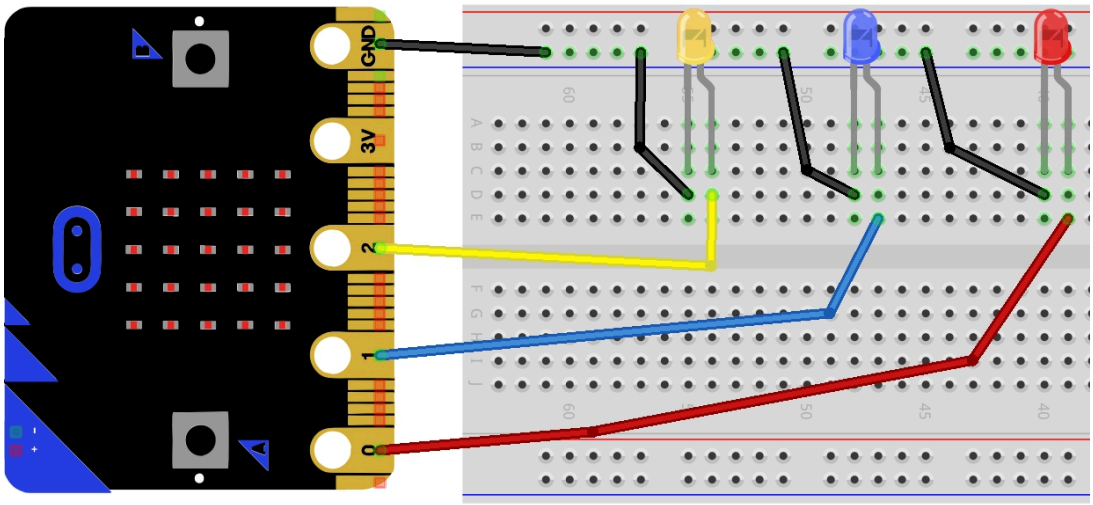


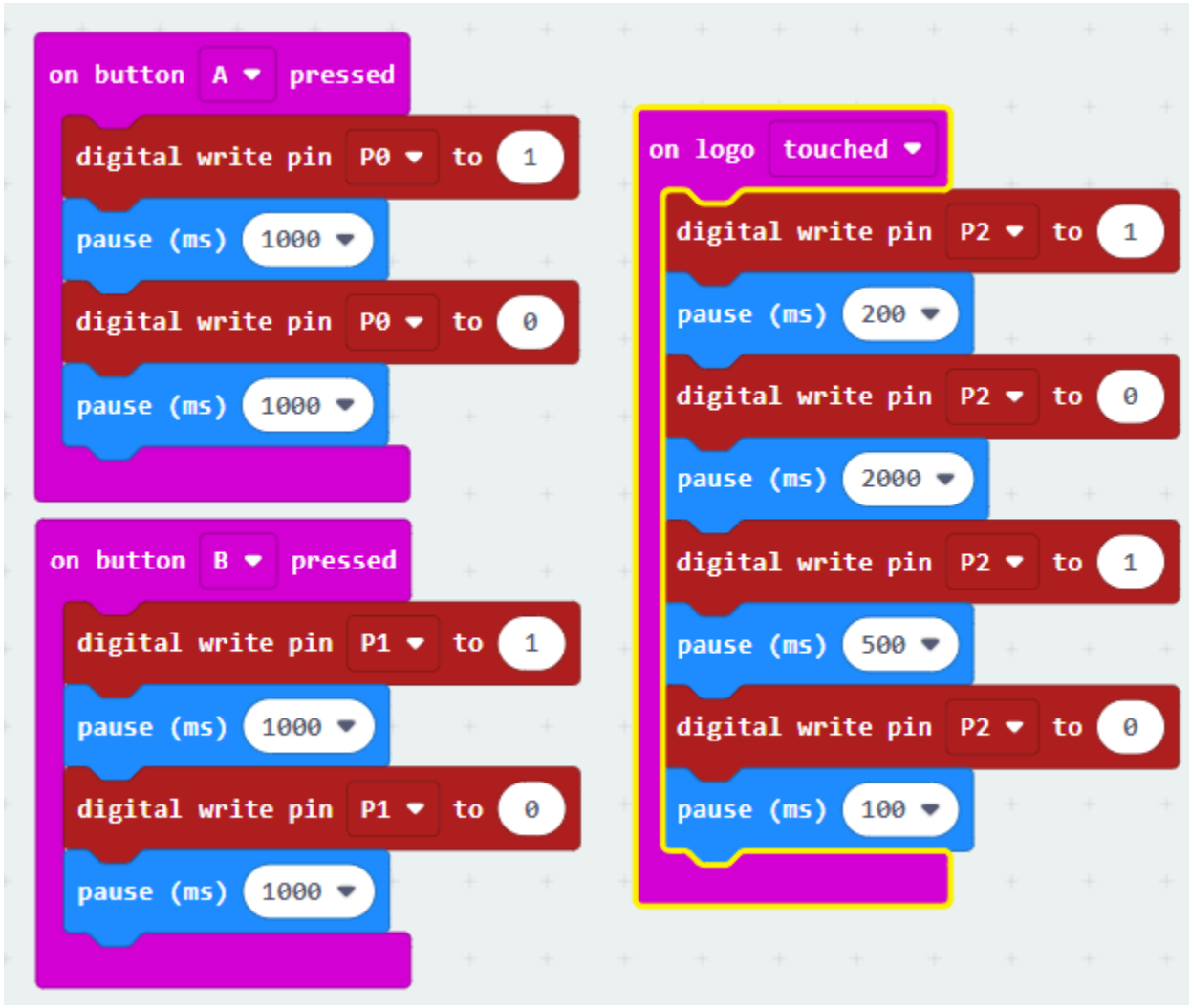
CHALLENGE 3: Using a breadboard and blinking 2 or 3 LEDs

1. Seeing how the single LED is wired to the breadboard, duplicate the pattern (negative lead to the ground rail, positive lead to a port/pin on the micro:bit)
2. Write code to blink each LED in a different way.

Things to consider:

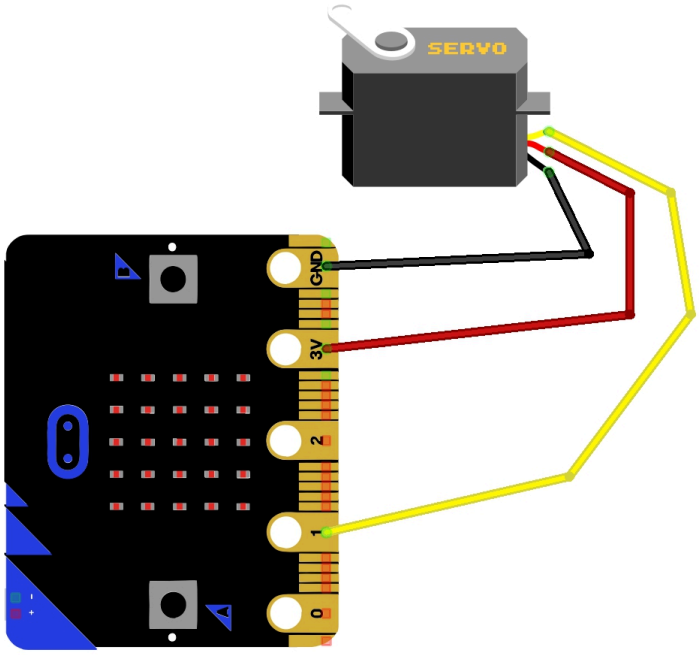
Remember what port/pin each LED is connected to and be sure your code for that LED matches!

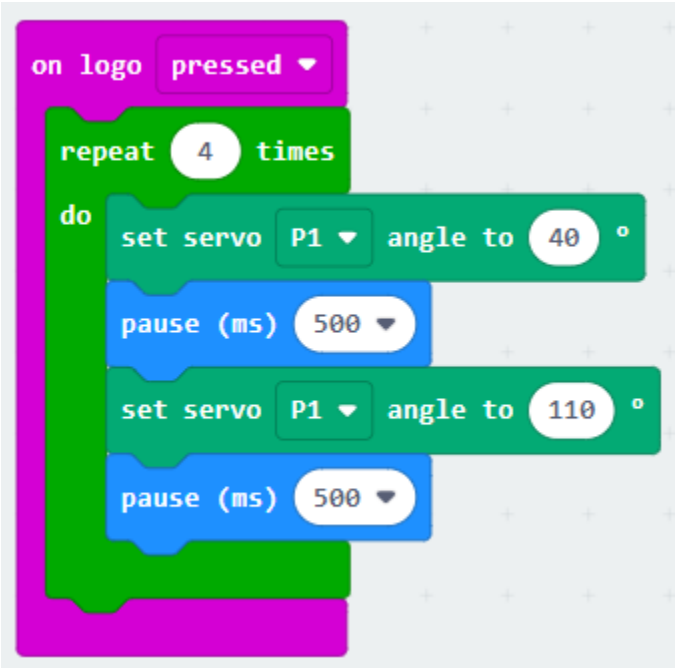




CHALLENGE 4: Connecting a servo motor to a micro:bit and making it move between 2 positions

1. Connect the GND port/pin to the ground(−) servo wire that is often black or brown.
2. Connect the 3V port/pin on the micro:bit to the positive(+) servo wire that is usually red.
3. Connect port/pin 1 to the servo signal wire, usually yellow or orange.
4. Next, in Makecode, add the extension code for the Servo.
5. Decide what 'event' will begin the servo moving. (this example uses the touch-logo)
6. Find the "set servo __angle to 90°" block.
    a. Set the pin to match the pin you connected it to - here it is p1
    b. Set the angle to whatever position you wish to start from. These servos are 0° to 180°.
7. If the objective is movement, you will now need a 2nd "set servo __angle to 90°" to return the servo to the 1st position.



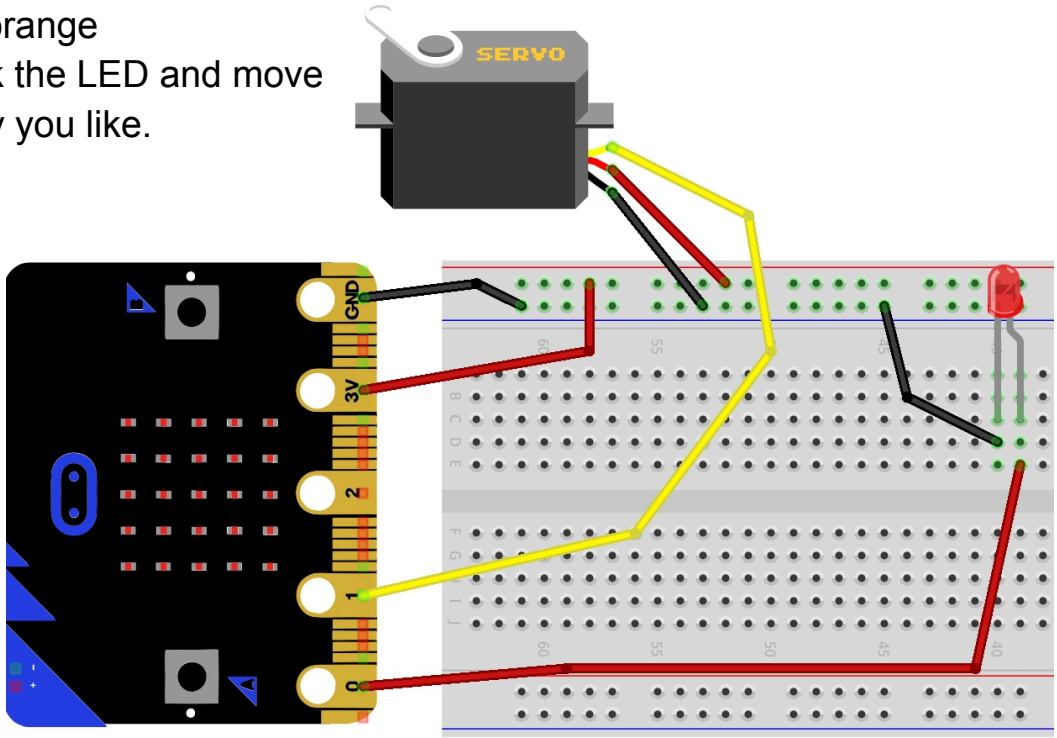


Things to consider:

The position servo needs power to run, (the red (+) wire) and a small amount of electricity from the board to tell it what position/degree to stop. This is the "signal" wire. Just like the LED, a pause is needed. This time is needed to allow the servo horn (the horn is what spins around) to get from one position to another. The amount of time needed will depend on the distance, you may need to experiment to find the time you need.

CHALLENGE 5: Using a breadboard: move servo and blink LED

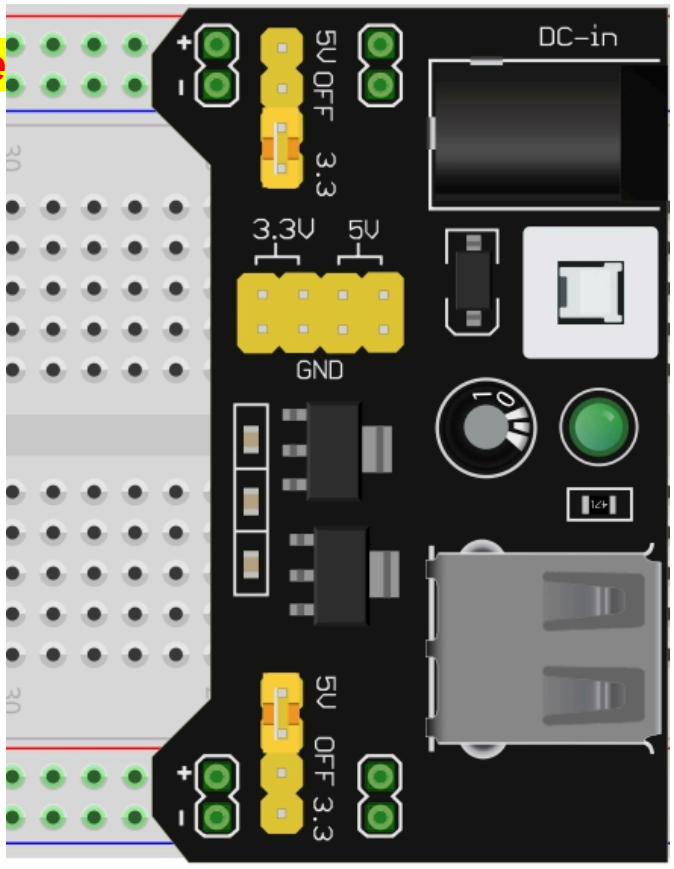
1. Connect the GND and the 3V ports/pins to the Breadboard to supply power and ground to multiple components.
    a. Connect the ground (GND) port on the micro:bit to the ground(−) RAIL on the breadboard
    b. Connect the 3V port/pin on the micro:bit to the positive(+) RAIL on the breadboard
2. Connect an LED:
    a. Use an alligator-to-male jumper to connect the ground (GND) port on the micro:bit to the ground(−) RAIL on the breadboard
    b. Place the LED on the breadboard remembering which is the longer (+) positive lead and which is the shorter (−) negative/ground lead
    c. Place a jumper in a breadboard hole that is in the SAME ROW as the negative(-) shorter lead of the LED
    d. Connect an alligator-to male jumper to port 0 and place the male end in a hole in the SAME ROW as the positive(+) lead on the LED
3. Connect a Servo:
    a. Place one end of a male-to-male jumper in the hole/pin at the end of the ground(−) servo wire that is often black or brown, place the other end in the ground(−) RAIL on the breadboard
    b. Place one end of a male-to-male jumper in the hole/pin at the end of the positive (+) servo wire that is usually red, then place the other end in the positive(+) RAIL on the breadboard.
    c. Using an alligator-to-male jumper, connect port/pin 1 to the servo signal wire, usually yellow or orange
    d. Write code to blink the LED and move the Servo any way you like.



Things to consider:

Remember w

CHALLENGE 6: Using a power-supply on a breadboard - sometimes 3 volts is just not enough…



VERY CAREFULLY - MATCH the positive and negative before pressing the power supply module onto the breadboard!!!!

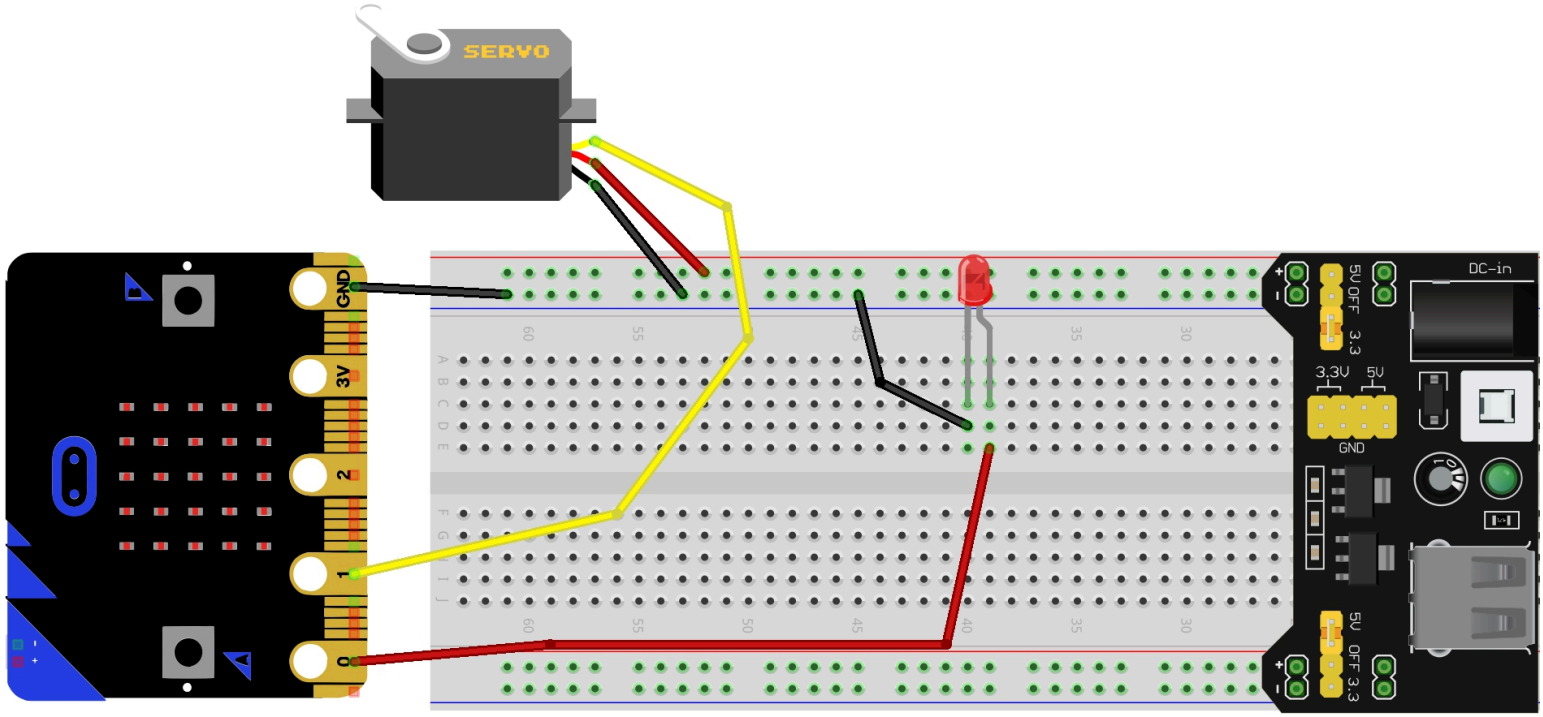
The positive in the **RED** rail, the negative in the **BLUE** rail

The pins that are spiky thick wires need to match up with the holes in the bread board, and pressed down into the holes on the rails of the breadboard. It can be very tricky so ask for help if you need it!

The wires can bend easily so be careful.

REMOVE the jumper from the 3V port/pin on the micro:bit. It is no longer needed since the power supply module will be providing the electricity to our components. Plug in your 9v power supply to the extension cord on the desk, plug the other end into the barrel jack on the power supply module.

Run your servo/LED code again to be sure everything is working!



CHALLENGE 7: Using a breadboard Breakout to access additional ports:

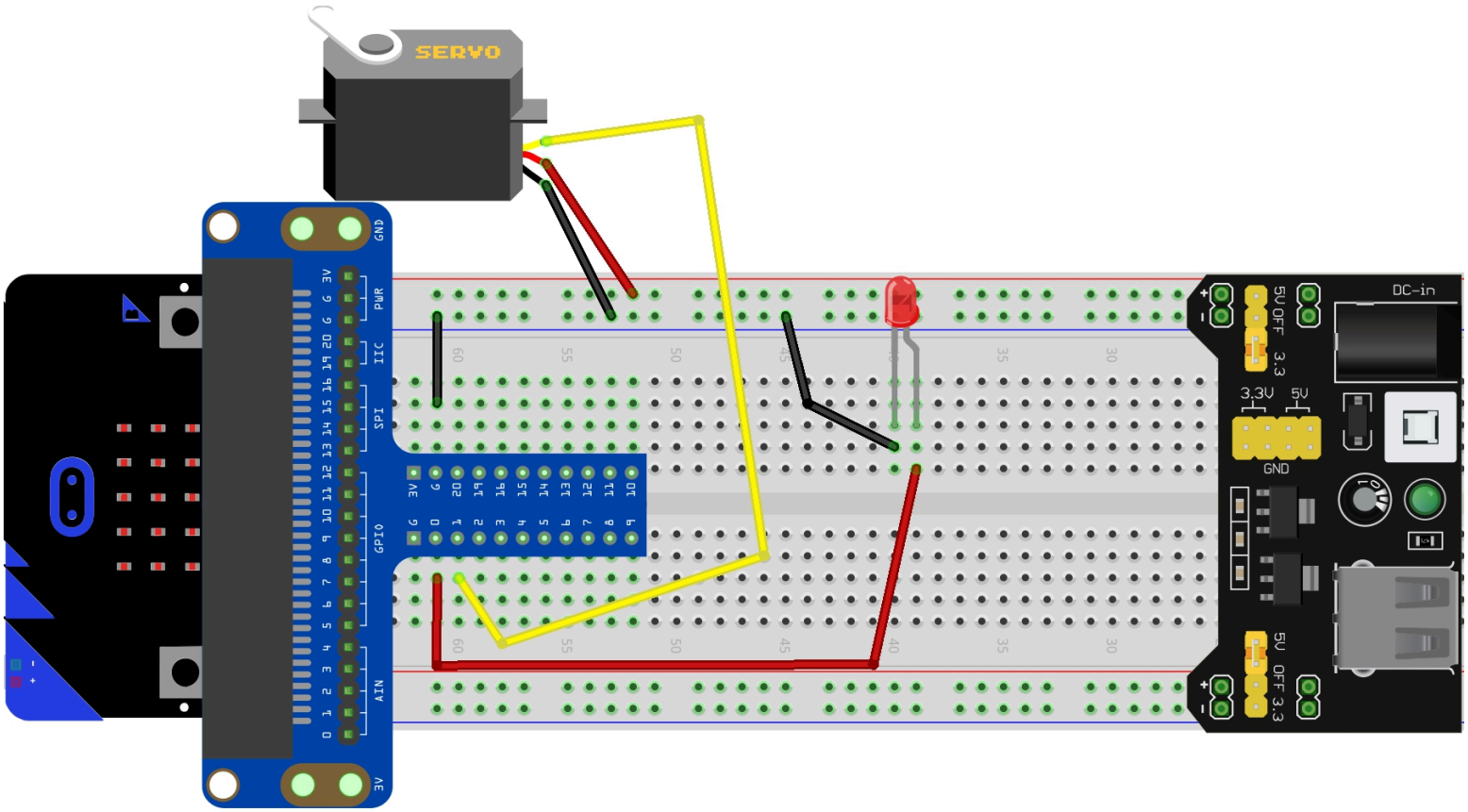
Breakout boards for the micro:bit come in different sizes, shape and configurations, but they all allow you to connect components to the tiny little ports that are the spaces between 0,1,2,3,3v and GND.

When using breakout boards, be very MINDFUL when connecting to be sure you are placing your jumpers in the correct breadboard pins that correspond to the labels on the breakout board. In this example, I am using the ports/pins labeled 0 and 1 for the LED and Servo, and there are 2 ports for GND - labeled simply with “G”.

NOTE: many of the ports are actually used by the array of LEDs on the face of the board, so not all 20 labeled ports are actually available to use. (but, if you need extra ports, the LED array can be turned off!)

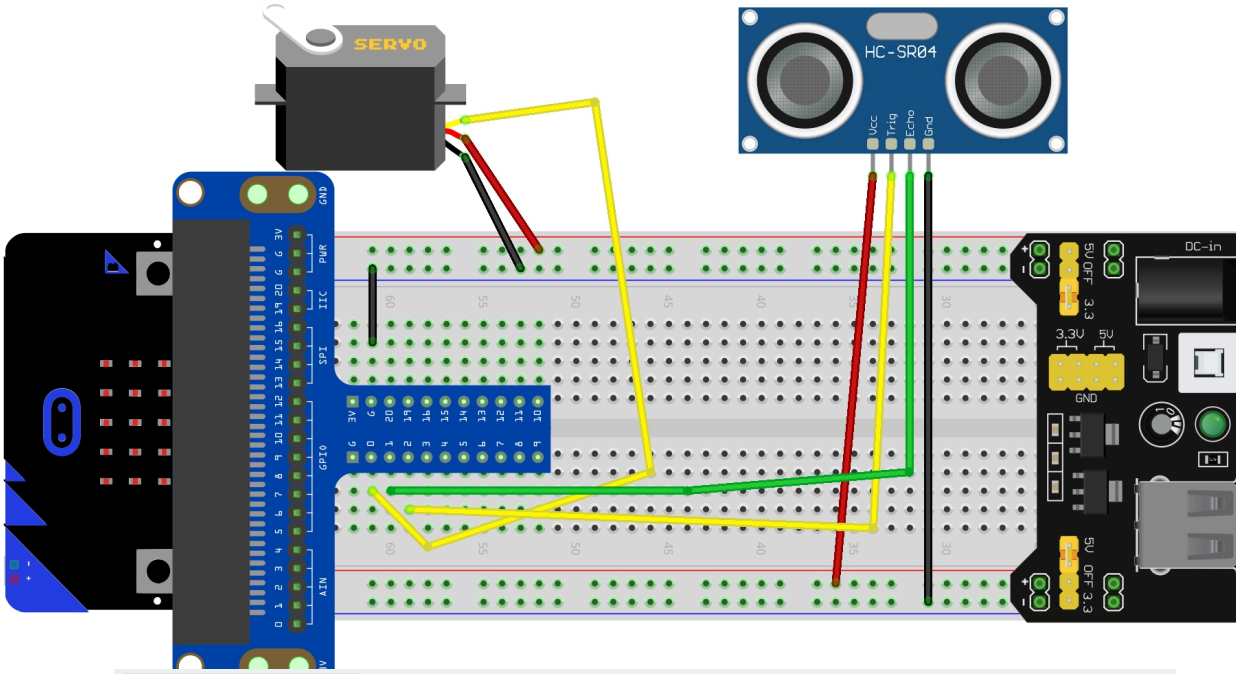
As with the power supply module, be careful pressing the pins into the holes on the breadboard.

Once the breakout board has been placed on your breadboard, re-wire your components with all male-to-male and connect to the rows that match the label for the port/pins. Test your existing servo/LED code again. Did you notice anything different?



CHALLENGE 8: Putting it to the test - using an HC-SR04 Ultrasonic Motion/Distance Sensor:

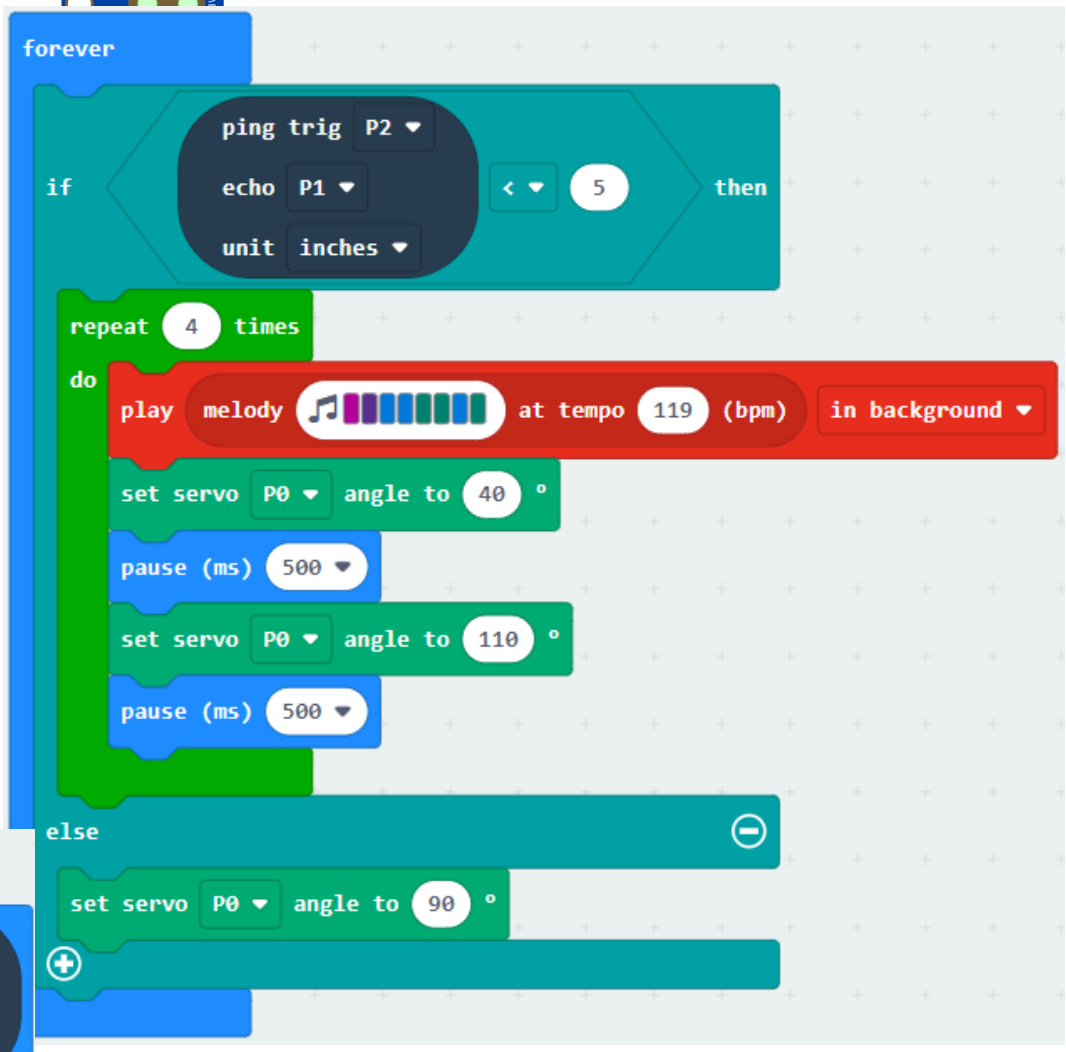
Extension:
"Sonar"
Use
"If-Then-Else"
checking sensor input - "is something close?" If yes, do this… if not, do that?

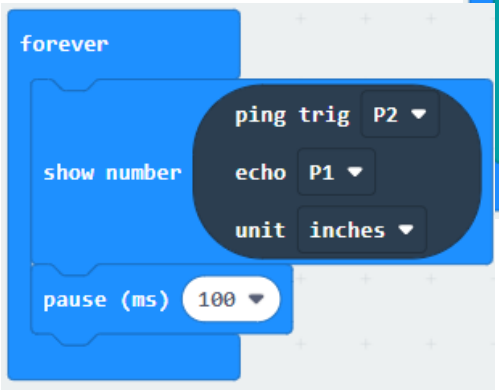


NOTE:
"trig" sends (triggers) the sound wave
"echo" waits to receive the sound wave back after it bounces off a surface.

Knowing the speed of sound - how long it takes to travel, the code can then figure out the distance of whatever the sound bounced off of.

You can choose inches or centimeters. You can also display the distance value.





Sure, go ahead, add more stuff!