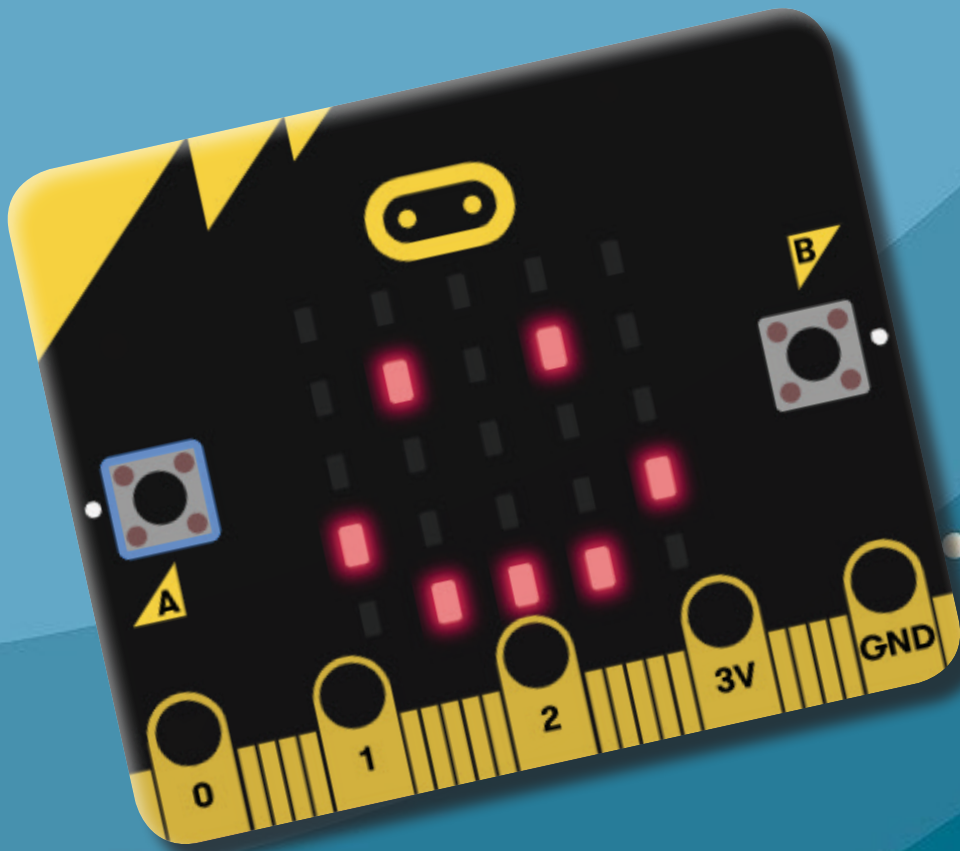


micro:bit Space Explorer!

centre for
computing
history _



WELCOME TO TWILIGHT@HOME

Hello and welcome to the Twilight@Home event.

In this activity, we will be looking at a device called the micro:bit. This is a pocket-sized codeable device that can be used to create fun projects. All the coding happens online. Also, for this activity you do not need the actual device as we will be using the simulator that is available on the coding website. For an introduction to the micro:bit, please watch this video (watch it up to the 13:21 mark for a quick introduction): <http://www.computinghistory.org.uk/det/62776/An-Introduction-to-Micro-bit-For-Children-with-No-Prior-Experience/>.

In this worksheet, we will start by creating an astronaut name badge and then move on to launching a rocket, which on the way sees a sky filled with twinkling stars.

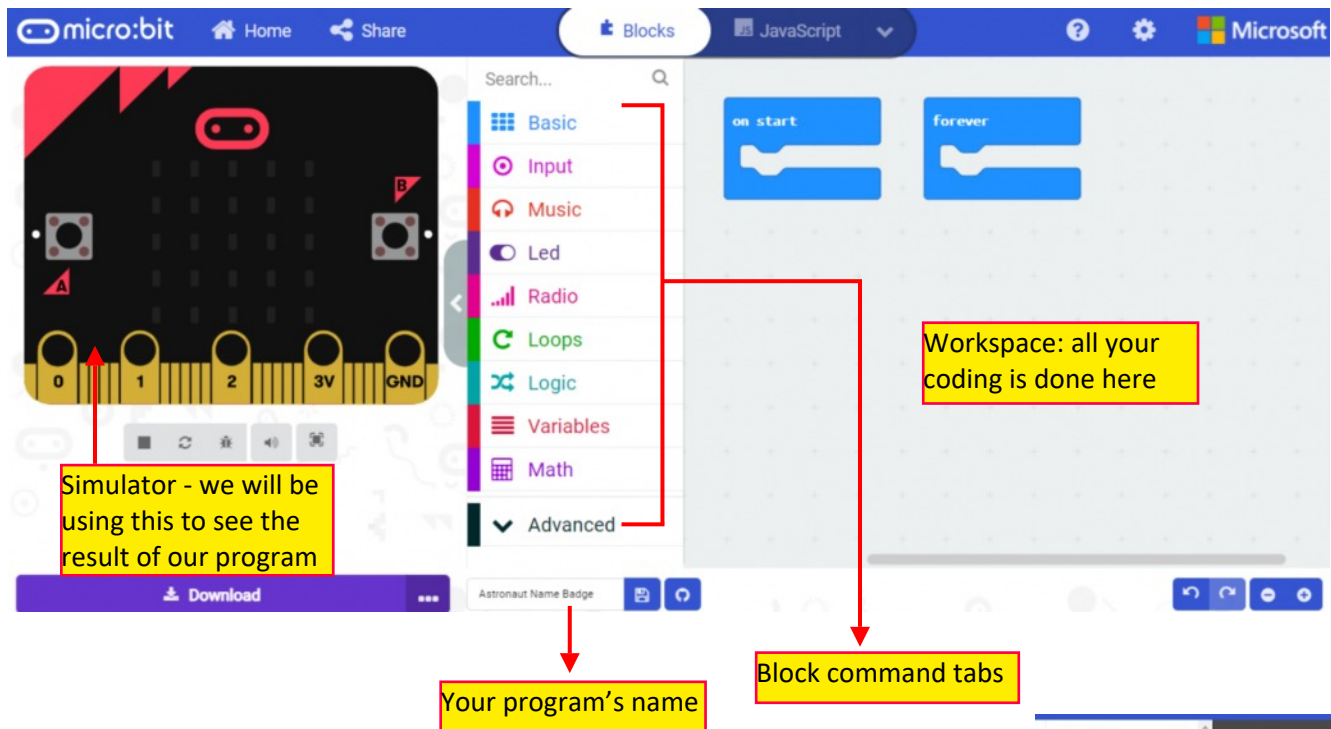
We hope you enjoy these activities. Happy coding!

HELLO micro:bit, I'M AN ASTRONAUT!

1. Make yourself a name badge, astronaut!

Go to makecode.microbit.org. Click on 'New Project'. It will ask you to give your project a name. Type a name of your choice (for example, you could call it 'Astronaut Name Badge'). Then, click on 'Create'.

You should see the Blocks Editor as shown below.



WRITE THE CODE: Time to write the program...


Click on the **Basic** tab and drag the **show string** block into the Workspace.

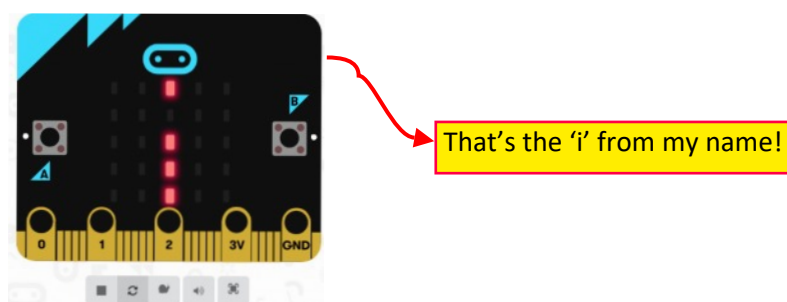
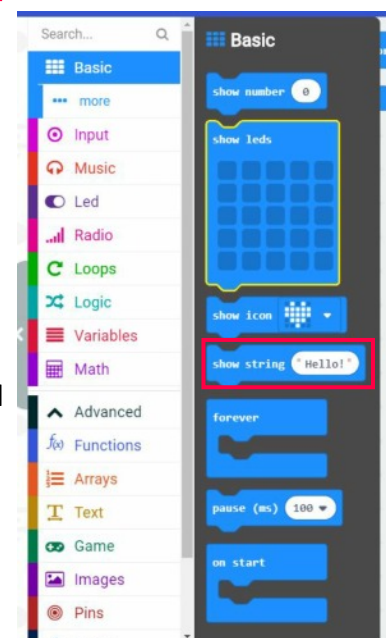
There are already two instruction blocks in the Workspace: **on start** and **forever**. These can be thought of as 'control' blocks (like in Scratch). They control when the instruction will be run.

Attach the **show string** block in the **forever** 'control' block.

Change the text from "Hello!" to your name: click inside the 'white space' and type in your name.

RUN THE CODE: The code should be running on the simulator.

You should see your name continuously scrolling on the simulator. If it doesn't appear, press the 'Start the simulator' icon  under the simulator.



What happens if you move the **show string** block to the **on start** block?

UP, UP AND AWAY

AIM:

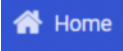
Create an animation that shows a rocket launching at the end of a countdown.

LEARNING POINTS:

This exercise shows you how to:

- Create an animation using the LEDs on the micro:bit;
- Use variables to store the countdown value; and
- Use the **IF-THEN** statement to decide when to take a particular action (in this case, launch the rocketship).

WRITE THE CODE:

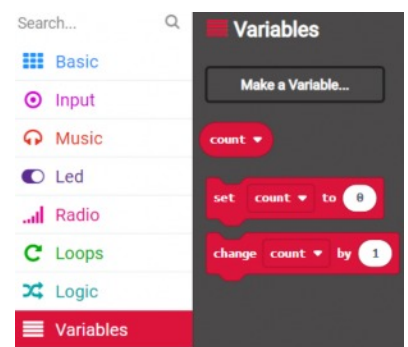
Click on the 'Home' icon  on the top left-hand corner of the screen. Click on 'New Project'. Type out the name of your project (for example, you could call the project 'Rocket').

Block command tabs you will need on Makecode:



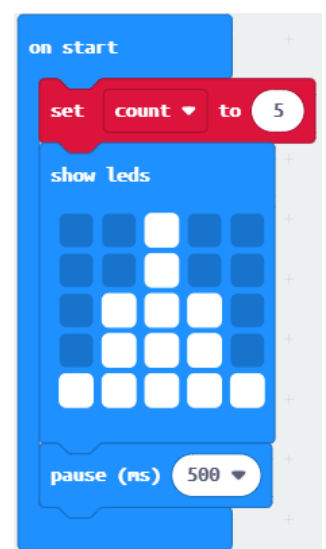
Task 1: Create a variable called *count* to store the countdown value.

- Go to the **Variables** tab and click on **Make a variable** button.
- In the 'New variable name' dialog, type in '*count*' and click on **Ok**.
- You will see ***count***, ***set count to*** and ***change count by*** blocks appear.



Task 2: Assign the countdown value and display the image of the stationary rocket.

- Drag the ***set count to*** block and attach it within the ***on start*** block. Click on the 0 displayed and change it to 5. [Our countdown starts from 5.]
- From the **Basic** tab, drag a ***show leds*** block and attach it under the ***set count to*** block. Now, click on the squares (which represent the LEDs on the micro:bit 'screen') to draw the image of a stationary rocket as shown in the image alongside. Every time you click on an empty square, it should light up.
- From the **Basic** tab, scroll down and drag a ***pause (ms)*** block and attach it under the ***show leds*** block. Click the 'arrow' next to the number displayed and choose **500ms**.



UP, UP AND AWAY


Now, we need to repeat certain actions continuously (in a '**forever**' loop):

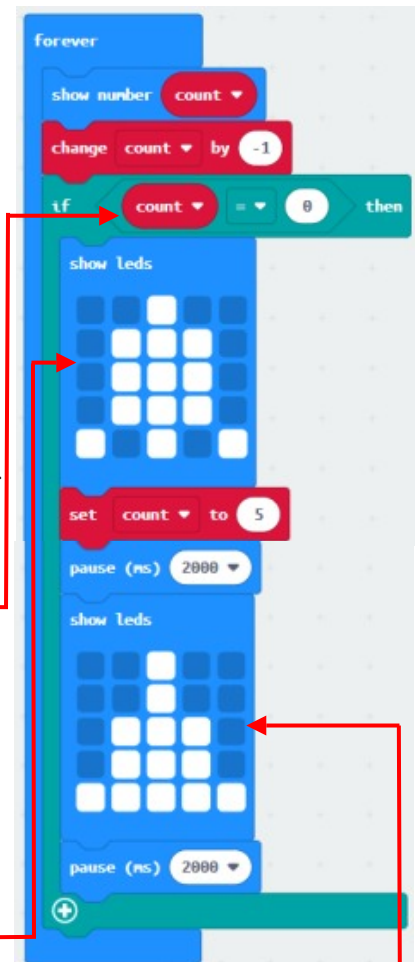
1. We want our countdown value to go down (5, 4, 3, 2, 1);
2. Once it reaches 0, the LED display should change to an image of a rocket taking off;
3. The countdown should be reset to 5 again;
4. And, the LED display should display the initial image of a stationary rocket.

Task 3: Show the value of count and start the countdown.

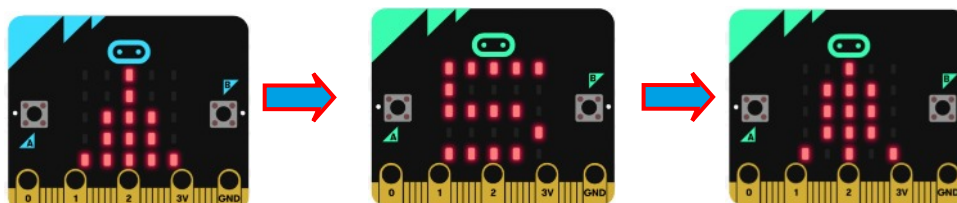
- Go to the **Basic** tab and drag the **show number** block and attach it within the **forever** block in the Workspace area.
- From the **Variables** tab, drag the **count** block and drop it in the **show number** block (where 0 is displayed).
- From the **Variables** tab, drag the **change count by** block and attach it under the **show number** block. Click on the 1 displayed and change it to -1. [The value of **count** goes down by 1 (or changes by -1).]

Task 4: IF the value of count reaches 0, THEN 'launch' the rocket and reset the count.

- Go to the **Logic** tab and drag the **if-then** block and attach it under the **change count by** block.
- From the **Logic** tab, drag the 'equal to' comparison block  and drop it in the **if-then** block where **true** is displayed.
- Drag the **count** block from the **Variables** tab and drop it in the comparison block on the left-hand side of the '=' sign (where 0 is displayed).
- From the **Basic** tab, drag a **show leds** block and attach it within the **if-then** block. Now, click on the squares to draw the image of a rocket taking off (as shown alongside).
- Under this, attach a **set count to** block (from the **Variables** tab). Click on the 0 displayed and change it to 5. [This resets the countdown variable.]
- From the **Basic** tab, drag a **pause (ms)** block and attach it under the **set count to** block. In the **pause** block, click the 'arrow' next to the number displayed and choose **2 seconds**.
- From the **Basic** tab, drag another **show leds** block and attach it to the **pause** block. Now click on the squares to draw the image of the stationary rocket as shown in the image above.
- Finally, add another **pause (ms)** block from the **basic** tab with a value of **2 seconds** under this.



RESULT:



TWINKLING STARS

Now that you're a little familiar with the micro:bit, the instructions for this exercise are going to be less detailed.

AIM:

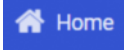
Create an animation that shows twinkling stars appear in the night sky.

LEARNING POINTS:

This exercise shows you how to:

- Create an animation using the LEDs on the micro:bit;
- Assign a random value to a variable;
- Control the status (on or off) and brightness of each LED on the micro:bit using its position;
- Use a **WHILE-DO** loop to repeat a set of actions as long as a condition is true.

WRITE THE CODE:

Click on the 'Home' icon  on the top left-hand corner of the screen. Click on 'New Project'. Type out the name of your project (for example, you could call the project 'Twinkling Stars').

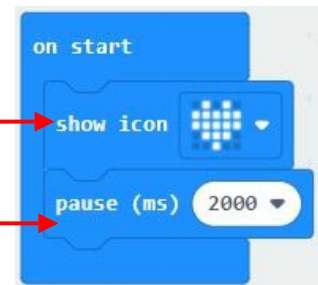
Blocks you will need on Makecode:



Write the code out in the Workspace area as shown in the images below (and on the next page).

Task 1: Start with a 'health check'.

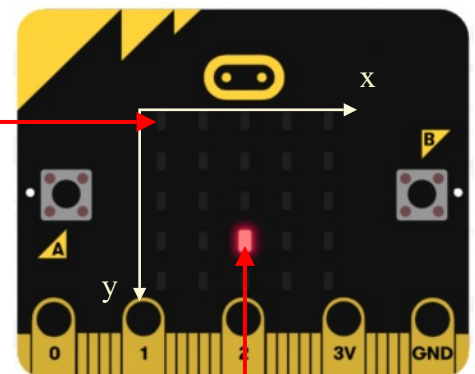
- Show an icon (I've used a Heart icon) for a few seconds. **NOTE:** This is good practice as it makes sure that the micro:bit is 'healthy' and working properly before your actual code starts running.



Find these blocks in the **Basic** tab.

Each LED on the micro:bit 'screen' can be identified by its position - that is, by a value on the x-axis or horizontal direction and a value on the y-axis or vertical direction.

- X-axis has values from 0 to 4.
- Y-axis also has values from 0 to 4.
- For example:
 - The top left-hand corner LED has a position (x=0, y=0).
 - The LED that is lit up has a position (x=2, y=3).



TWINKLING STARS

Task 2: Create the variables you need.

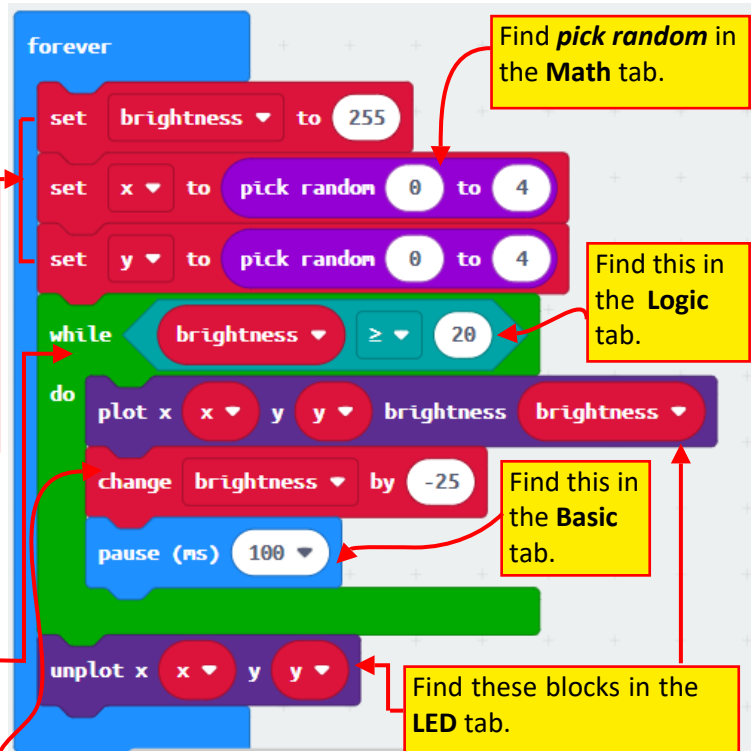
- Create three variables and call them **x**, **y** and **brightness**. **x** records the X-axis position of an LED, **y** records the Y-axis position of an LED and **brightness** records the brightness of an LED. [You can remind yourself how we did this with the count variable in the rocket exercise.]
- Set **brightness** to 255 [this is the maximum value of brightness possible.]
- Set **x** and **y** to random values between 0 and 4. [Every time the code runs, a random LED gets picked.]

Find these blocks in the **Variables** tab. When you drag a **set to** block, click on the arrow next to the variable name and you will be able to choose between **x**, **y** and **brightness**.

NOTE: You can also duplicate a block by right-clicking on it and choosing 'Duplicate'.

Find the **while-do** block in the **Loop** tab.

Find this block in the **Variables** tab.



Task 3: Light up the randomly chosen LED and keep reducing its brightness until the value falls below a certain level.

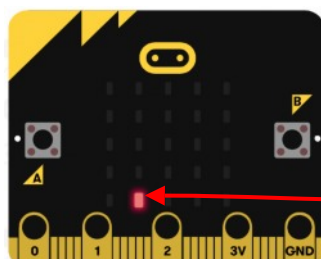
- Use a **while-do** loop to check if **brightness** is above a certain level [for example, ≥ 20].
- As long as the above condition is true, do the following:
 - Using the **plot x y brightness** block (find this in the **LED** tab), switch on the randomly chosen LED and set the brightness to the variable **brightness**.
 - Using the **change by** block (find this in the **Variables** tab), reduce **brightness** by 25.
 - Using the **pause (ms)** block (find this in the **Basic** tab), pause for 100ms.

Task 4: Switch off the lit-up LED to get the flashing effect.

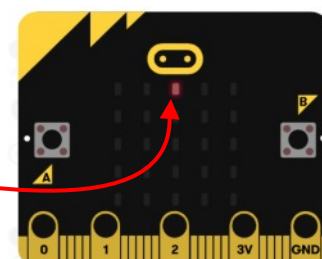
- Using the **unplot x y** block (find this in the **LED** tab), switch off the random LED that was switch on.

RESULT:

LED chosen:
x=1, y=4



Notice the difference in brightness of the two LEDs



LED chosen:
x=2, y=0