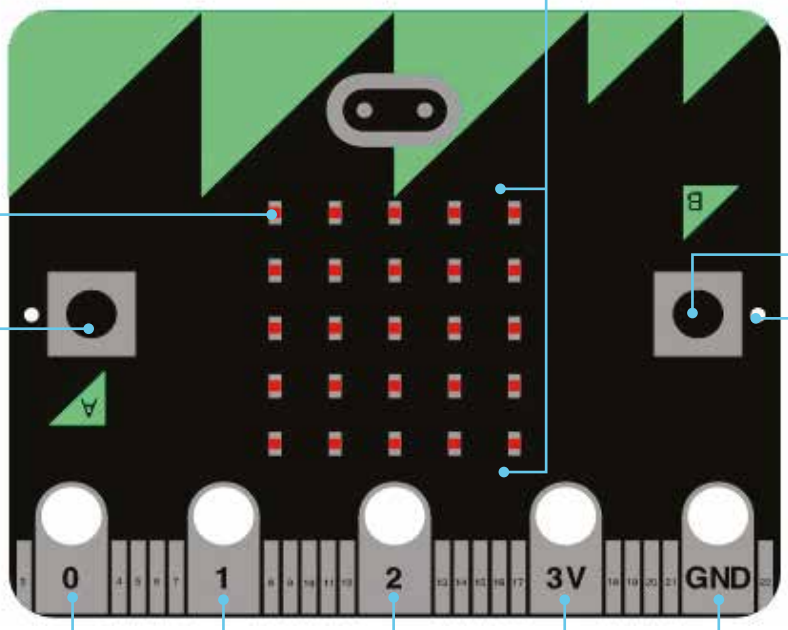
The BBC micro:bit: What is it designed to do?

The BBC micro:bit is a very simple computer. A computer is a machine that accepts input, processes this according to stored instructions and then produces output. All three of these elements are present on the BBC micro:bit's printed circuit board.

Front of board

LED
Coordinates start at (0,0) in top left-hand corner. In computing, displays start at the top left-hand corner so, in coding terms, this is (0,0). This is different from mathematics and graphs where (0,0) is the bottom left corner. It is important to note this is also relative, so if the screen rotates (0,0) is still the top left corner of the screen. See Lesson 4 for the use of coordinates in the Catch the egg game.

LED MATRIX
5 × 5 array of light emitting diodes (LEDs), which can each be set to on / off. The brightness of the set of LEDs as a whole can also be controlled.



BUTTON B
See Button A.

HOLES
Holes for sewing, mounting and hanging.

BUTTON A
A form of input. The BBC micro:bit detects when this button is being pressed. This is a push-to-make switch (pressing it completes an electrical circuit).

PINS P0, P1, P2
Pins for attaching external sensors, like thermometers or moisture detectors, and actuators, like turning a motor on, so students can build projects with them like a plant watering alarm. Can be either input or output and either digital or analogue.

3V AND GND
Enable a user to power an external device, like a motor, using the battery or USB. They also enable capacitive touch (using an object as a switch).

Back of board

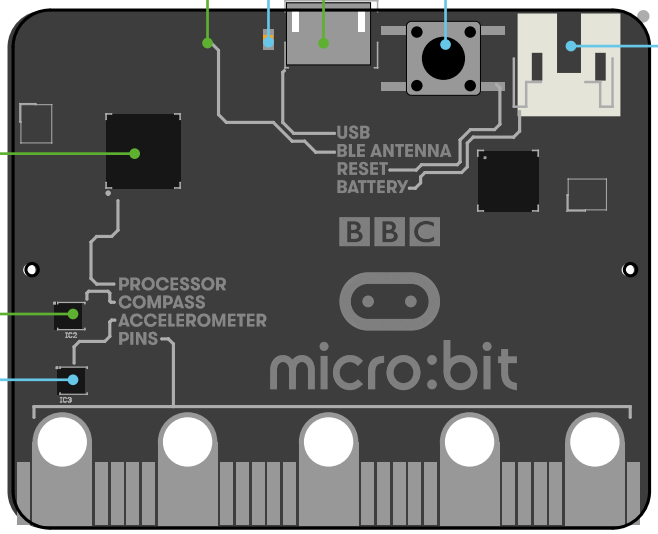
BLUETOOTH LOW ENERGY ANTENNA
A messaging service, built for the Internet of Things, so devices can talk to each other. The BBC micro:bit will be a peripheral device and it can talk to a central device like a smartphone or tablet (or a laptop that has BLE). This means the BBC micro:bit can send signals to and receive signals from a central device. BLE will be used to 'flash' new programs onto the BBC micro:bit and to allow the BBC micro:bit to communicate with a computer or an internet connection.

USB PLUG
Programs can be downloaded from Windows and Macs onto the BBC micro:bit via a USB data connection. The USB connects the BBC micro:bit to a computer. This means the BBC micro:bit can send data to and receive data from the computer. The USB will be used to 'flash' new programs onto the BBC micro:bit and to allow the BBC micro:bit to communicate with a computer or an internet connection.

STATUS LED
Flashes yellow when the system wants to tell the user something has happened.

BUTTON R
System button, which has various uses. Has to be pressed to 'flash' new code onto the device over BLE.

PROCESSOR
All the BBC micro:bit's programs and any data are stored on the small silicon-chip micro-controller. This tiny chip designed by ARM has 128kB flash memory and 16kB RAM memory; a tiny fraction of the memory on a smartphone.



BATTERY
This socket connects the external battery pack (containing two AAA batteries) to the board. The battery pack is attached physically to the board with a Velcro patch.

COMPASS
A sensor to detect magnetic fields, like the Earth's, allowing the direction of the BBC micro:bit to be determined and converted to a digital form that can be used in BBC micro:bit programs. Output from the compass is degrees.

ACCELEROMETER
Converts analogue information about how quickly the BBC micro:bit's speed changes to a digital form that can be used in BBC micro:bit programs. Output from the accelerometer is in milli-g. Allows the BBC micro:bit to be used to control movement of on-screen characters such as Kodu.

A note about ARM

ARM designs the processors for most mobile phones and embedded systems (such as smart thermostats, car engine controllers and the processors inside digital cameras), and was founded by members of the original BBC Micro team!

BBC micro:bit is based on ARM's mbed platform for embedded systems, but programming the BBC micro:bit is very straightforward.

A note about machine code

Machine code is the language the CPU (central processing unit) of a computer understands, but it isn't very readable by humans as it is made up of numbers.

Machine code is known as a low level language. High level languages, such as Microsoft Block Editor or Microsoft Touch Develop, are readable/understandable by humans. A program written in a high level language, like Microsoft Touch Develop, has to be compiled (translated) into machine code that the processor 'understands' (see page 10).

Lesson 1: Digital key chain

Programming a Minecraft Creeper face using the image editor within Microsoft Touch Develop

Outcome

Display of a Creeper face (similar to the character seen in Minecraft) on the BBC micro:bit LED display:

- By default, all of the LEDs (or lights) are off.
- There will be a single state (Minecraft Creeper face).
- The image will turn off after 3 seconds.

Tutorials

For a guided tutorial go to
www.microbit.co.uk/td/tutorials/digital-key-chain

Decomposing the problem

This lesson can be decomposed into four parts:

1. Design how our single state will look (which LEDs will be switched on to display our Creeper face).
2. Use the image editor to turn on the required LEDs.
3. Create a timer to pause the image for 3 seconds.
4. Reset the display to its original state: OFF.

Design how each **state** will look

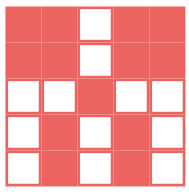
Before we start to code, we need to plan what our single state will look like.

01

Draw a 5×5 grid and colour in the boxes to show what the Minecraft Creeper face will look like.

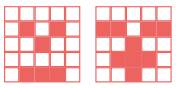
You don't have to program a Creeper face.
The image could be anything you like.

Key
LED on = ■
LED off = □



CREEPER FACE

Why not try out different images?



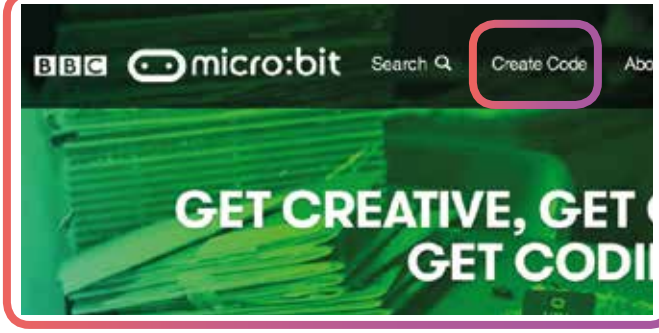
Use the image editor to **turn on** the required LEDs

We need to specify which LEDs will be ON to display the Creeper face.

02

Start by opening a new browser window and typing **www.microbit.co.uk** in the address bar.

Click on **Create Code**. In **Microsoft Block Editor**, click **New project**. Type in a name for your script, such as **`Creeper`**. Click on **create**.



03

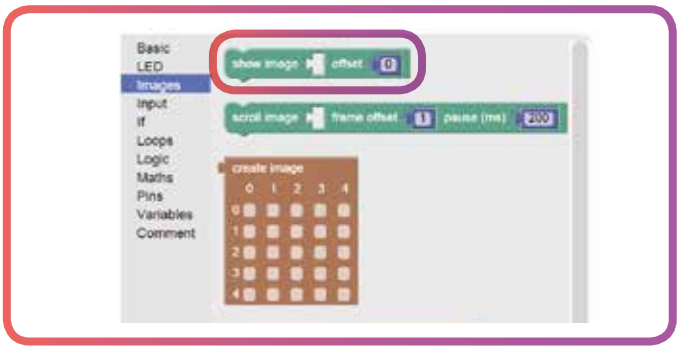
A blank coding environment will appear (see picture to the right).

Select the **Images** button from the menu on the left.

04

The **Images** section includes blocks that control the creation and display of an image on the BBC micro:bit through LEDs.

Select the **show image** block.



05

You will notice that an offset value of **0** is displayed.

Changing this allows you to show your image in different positions on the BBC micro:bit display.

06

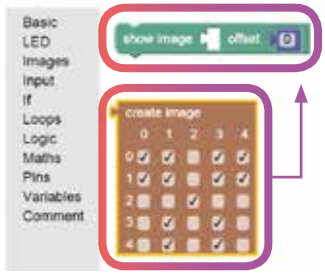
We now want to select which LEDs will be ON for our Creeper face.

Select the **Images** button then the **create image** block. Tick the boxes in the block to make the shape of the Creeper face, as shown in the image.

07

Drag the **create image** block into the empty position on the **show image** block.

This will make sure that the Creeper face appears when you press the **run** button.



It's important that we test our programs regularly. This allows us to debug the program and fix any errors.

08

Press the **run** button to test your program.
What does it look like on the simulator?
If it doesn't work as expected, go back and try to find and correct the problem.

Create a timer to **pause** for 3 seconds

To display the Creeper face for a short period of time, we need to add a timer.

09

From the **Basic** menu, select the **pause** block. The BBC micro:bit uses milliseconds as input, so 1000 is equivalent to 1 second.

We want to pause for 3 seconds, so change the number to **3000**.

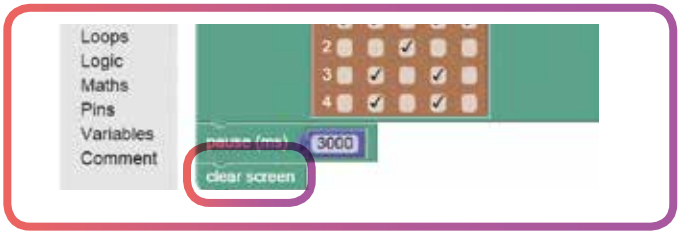
Drag the block upwards so it snaps into place below the **show image** block.

Reset the display to its original state: **OFF**

To finish our program, we're going to turn all of the LEDs off. This will help to prolong the battery life of the BBC micro:bit.

10

Click on the **LED** menu. Select the **clear screen** block and snap it under the **pause** block. This will make sure that all of the LEDs are turned off after the Creeper face has displayed for 3 seconds.



11

You should now have a finished program which will display a Creeper face!

Do your own thing!

- Change the pattern in the **create image** block to show your own design.
- Instead of clearing the display, add another **show image** and **pause** block to create a simple two-state animation. Can you experiment with the brightness of the Creeper image between face changes?

A solution for the complete digital key chain code can be found on page 31. The working code can be found at **www.microbit.co.uk/start-guide/solutions/digital-key-chain**.