Environment Detective: Exploring Our World with micro:bit

Microbit | Beginner | 45 Mins

1 Overview

Ever wondered how technology can help us understand the world around us?

In this project, you are going to become an environment detective using your micro:bit and its built-in sensors.

Just like your senses help you explore the world, the micro:bit can tell you:

- How warm or cold it is using its temperature sensor
- How bright or dark it is using its light sensor
- Which direction you are facing using its compass

You will write Python code to make the readings appear on the micro:bit's LED matrix.

What you'll learn

- Read the micro:bit's temperature sensor
- Display numbers, pictures and letters on the LED matrix
- Detect light levels and make the micro:bit react
- Use the compass to find a direction
- Use Python if statements to make decisions
- Use buttons, shake gestures and variables in your code

Are you ready? Let's begin our detective adventure!

2 What you'll need

🔧 What you'll need

- micro:bit
- micro:bit Python editor
- Micro USB cable
- Battery pack (optional)

3 Key words

💡 Key words

micro:bit

The micro:bit is a tiny programmable computer, about the size of a matchbox, made for learning to code. It has two buttons, a grid of little red LED lights and built-in sensors that can detect movement, light and temperature.

A bit like a pocket-sized computer you can program yourself, a micro:bit can become a step counter, a thermometer or even a simple game.

Temperature sensor

A temperature sensor is a little part that measures how warm or cold it is around it, then turns that into a number your code can read. As the room heats up or cools down, the number changes too.

Like the thermometer that tells you if you've got a temperature when you're poorly, a temperature sensor measures the warmth around it and gives your code a number to work with.

Light Sensor

A light sensor is a little part that measures how bright or dark it is around it, then turns that into a number your code can read. As the light changes, so does the number.

Like the automatic headlights on a car that switch on when it gets dark, a light sensor notices the light dropping and lets your code react.

Compass

A compass senses which way you are facing and turns it into a number your code can read, from 0 all the way round to 359. North is 0, east is 90, south is 180 and west is 270.

Like the little arrow on a map app that swings round as you turn, a compass tells your code which way you are pointing.

LED Matrix

The LED matrix is the grid of little red lights on the front of the micro:bit. By switching them on and off in patterns, your code can show numbers, letters and tiny pictures.

Like a mini scoreboard made of dots, the LED matrix lights up to spell out your step count.

Variable

A variable is something that can change. Think of it like a labelled box: you pop something inside, and the label tells you what it is. You can swap what's inside whenever you like.

A variable called score might start at 0, then change to 1, 2 and 3 as you collect points in a game.

If statement

An if statement checks whether something is true, and only runs its instructions when it is. It's how code makes decisions.

"If it's cold, put on a coat." If it's warm, you skip the coat, the action only happens when the check is true.

Comparison Operators

A comparison operator checks how two things measure up, for example whether they are equal or one is bigger than the other. It gives back a simple true or false answer.

Asking "is 6 == 6?" gives back true, but "is 6 == 2?" gives back false, so your code knows whether the two numbers match.

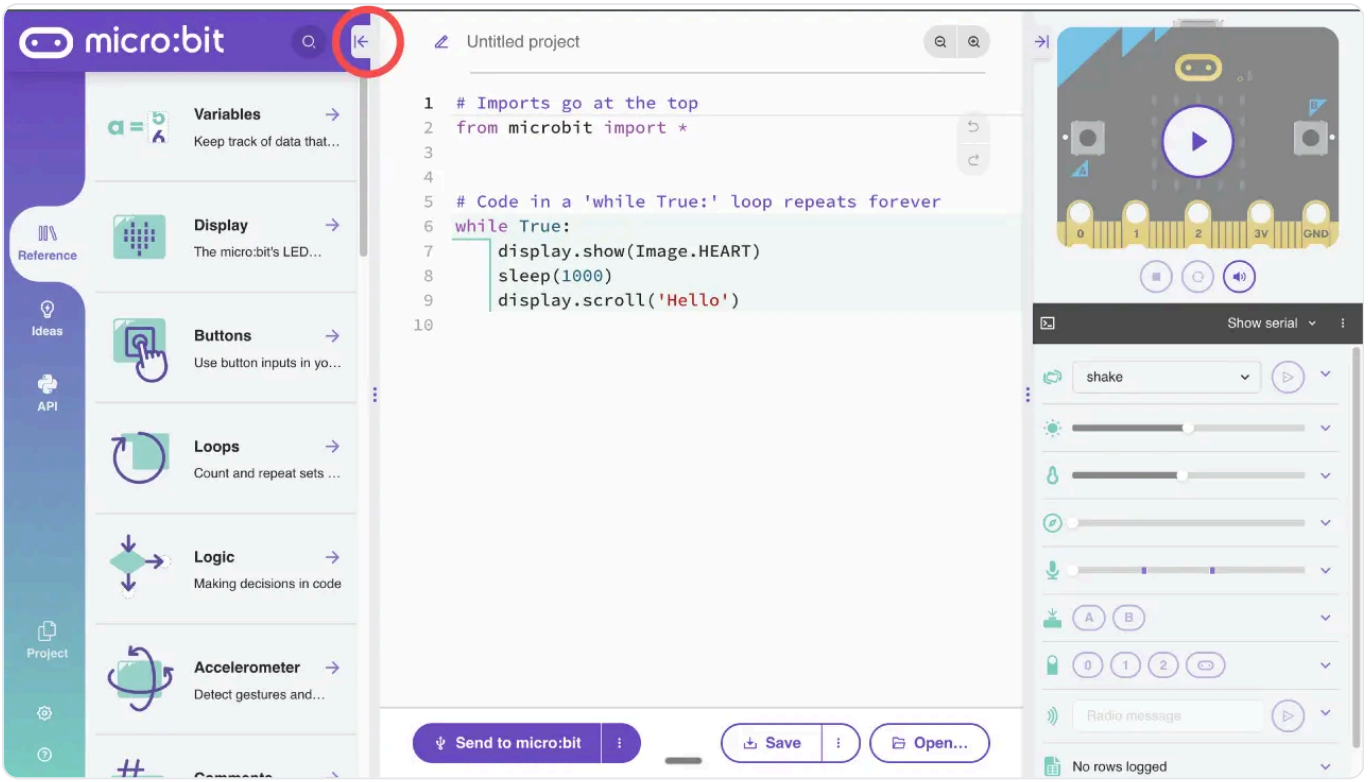
4 Set up the Python editor

First, let’s prepare the micro:bit Python editor.

1. Open your favourite web browser. Google Chrome or Microsoft Edge works well.
2. Go to python.microbit.org.
3. Close the panel on the left by selecting the arrow pointing left.
4. Find the starter code in the main coding area.
5. Delete the code from line 5 to line 9.
6. Keep this first line:

```
from microbit import *
```

Your editor should now be ready for your Environment Detective code.



`from microbit import *` gives your Python program access to the micro:bit's display, buttons, sensors and other tools.

5 Prepare your Python code

We need to prepare the compass and create a loop that keeps checking the sensors.

Under `from microbit import *`, type:

```
compass.calibrate()
```

```
while True:
```

Press **Enter** after `while True:`.

The cursor should move inwards automatically. This space at the beginning of the line is called an **indent**.

```
# Imports go at the top
from microbit import *
compass.calibrate()

while True:

```

🔁 `while True:` creates a loop. Everything indented underneath it will keep running while the micro:bit is switched on.

⌨️ **Python punctuation matters**

Remember to include the colon at the end of `while True:`. Python uses colons and indentation to work out which instructions belong together.

6 Sense the temperature

First, we will make the micro:bit show the temperature when you shake it.

Underneath `while True:`, type:

```
    if accelerometer.was_gesture("shake"):
        display.scroll(temperature())
        sleep(2000)
        display.clear()
```

Use four spaces for each level of indentation.

Your code should look like this:

```
# Imports go at the top
from microbit import *
compass.calibrate()

while True:
    if accelerometer.was_gesture("shake"):
        display.scroll(temperature())
        sleep(2000)
        display.clear()

```

Here's what this bit of code is doing:

- `accelerometer.was_gesture("shake")` checks whether the micro:bit was shaken.
- `display.scroll(temperature())` reads and displays the temperature.
- `sleep(2000)` waits for two seconds.
- `display.clear()` clears the LED matrix.

🌡️ The micro:bit displays temperature in degrees Celsius.

💡 The number inside `sleep()` is measured in milliseconds. There are 1,000 milliseconds in one second, so `sleep(2000)` waits for two seconds.

7 Sense the light level

Next, we will use button A to check whether the room is dark.

Make sure the new `if` line is aligned with the temperature `if` line.

Add this code underneath the temperature section:

```
    if button_a.was_pressed():
        if display.read_light_level() < 100:
            display.show(Image.HAPPY)
        else:
            display.clear()
        sleep(500)
```

Your light-sensing code should look like this:

```
# Imports go at the top
from microbit import *
compass.calibrate()

while True:
    if accelerometer.was_gesture("shake"):
        display.scroll(temperature())
        sleep(2000)
        display.clear()
    if button_a.was_pressed():
        if display.read_light_level() < 100:
            display.show(Image.HAPPY)
        else:
            display.clear()
        sleep(500)

```

When you press button A:

- The micro:bit reads the light level.
- If the reading is below `100`, it shows a happy face.
- Otherwise, it clears the display.
- It waits for half a second before checking again.

The code uses an if statement and a comparison operator to make this decision.

🔦 Cover the micro:bit with your hand and press button A. Then move it somewhere brighter and try again.

💡 The `<` symbol means **smaller than**. The line `display.read_light_level() < 100` asks whether the light reading is smaller than 100.

8 Build the compass code

Now we will use button B to display the compass direction as a number.

Add this code underneath the button A section:

```
    if button_b.was_pressed():
        display.scroll(compass.heading())
        sleep(500)
```

Your code should look like this:

```
# Imports go at the top
from microbit import *
compass.calibrate()

while True:
    if accelerometer.was_gesture("shake"):
        display.scroll(temperature())
        sleep(2000)
        display.clear()
    if button_a.was_pressed():
        if display.read_light_level() < 100:
            display.show(Image.HAPPY)
        else:
            display.clear()
        sleep(500)
    if button_b.was_pressed():
        display.show(compass.heading())
    sleep(500)
```

When you press button B, the micro:bit will display a number between `0` and `359`.

Direction	Compass heading
North	0
East	90
South	180
West	270

The compass works like a circle. After 359 degrees, it returns to 0 degrees.

Test this version before moving on. Once it works, we will replace it with code that displays letters instead.

9 Show compass letters

Numbers are useful, but letters are easier to recognise quickly.

Remove this version of the button B code:

```
    if button_b.was_pressed():
        display.scroll(compass.heading())
        sleep(500)
```

Replace it with:

```
    if button_b.was_pressed():
        direction = compass.heading()

        if direction > 315 or direction < 45:
            display.show("N")
        elif direction < 135:
            display.show("E")
        elif direction < 225:
            display.show("S")
        else:
            display.show("W")

        sleep(500)
```

Your finished compass code should look like this:

```
# Imports go at the top
from microbit import *
compass.calibrate()

while True:
    if accelerometer.was_gesture("shake"):
        display.scroll(temperature())
        sleep(2000)
        display.clear()
    if button_a.was_pressed():
        if display.read_light_level() < 100:
            display.show(Image.HAPPY)
        else:
            display.clear()
        sleep(500)
    if button_b.was_pressed():
        direction = compass.heading()
        if direction > 315 or direction < 45:
            display.show("N")
        elif direction < 135:
            display.show("E")
        elif direction < 255:
            display.show("S")
        else:
            display.show("W")
    sleep(500)

```

How the code works

This line creates a variable called `direction`:

`direction = compass.heading()`

It stores the current compass heading as a number.

The code then checks that number:

- More than `315` or less than `45` means north
- Less than `135` means east
- Less than `225` means south
- Anything else means west

> 🍕 **How your code decides**
>
> Imagine the compass is a pizza cut into four slices. The code checks which slice the compass number has landed in, then shows the letter for that direction.

10 Check the finished code

Your complete Python program should now look like this:

```
from microbit import *

compass.calibrate()

while True:
    if accelerometer.was_gesture("shake"):
        display.scroll(temperature())
        sleep(2000)
        display.clear()

    if button_a.was_pressed():
        if display.read_light_level() < 100:
            display.show(Image.HAPPY)
        else:
            display.clear()
        sleep(500)

    if button_b.was_pressed():
        direction = compass.heading()

        if direction > 315 or direction < 45:
            display.show("N")
```

```
        elif direction < 135:
            display.show("E")
        elif direction < 225:
            display.show("S")
        else:
            display.show("W")

        sleep(500)
```

Before you run it, check:

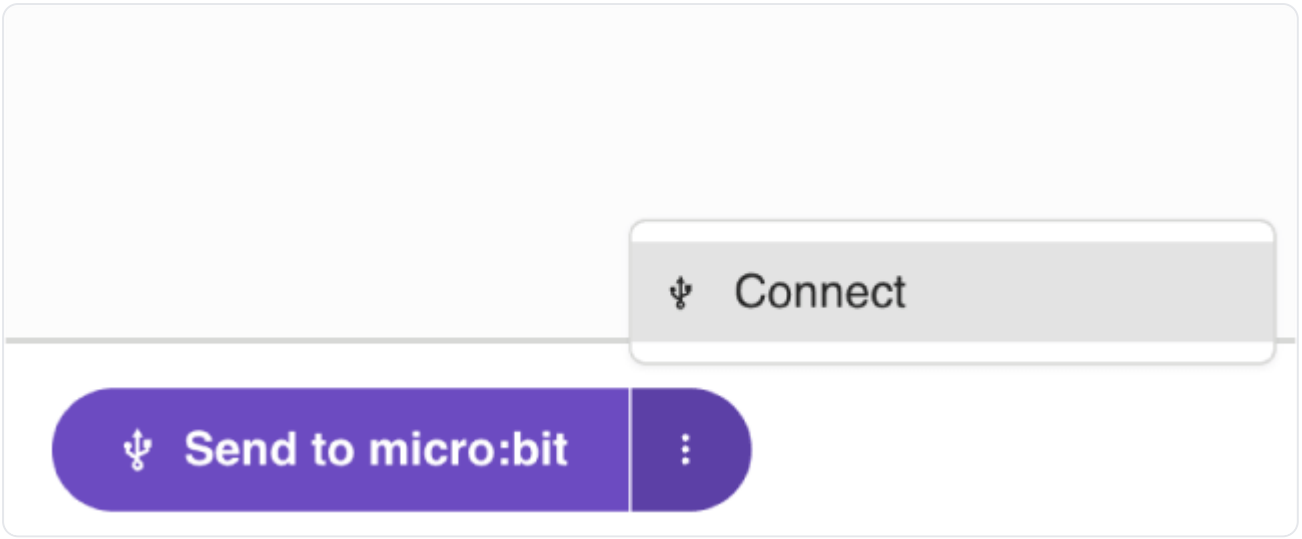
- Every `if`, `elif`, `else` and `while` line ends with a colon.
- Each section has the correct indentation.
- Words such as `Image.HAPPY` use the correct capital letters.
- Every opening bracket has a matching closing bracket.
- The words inside quotation marks are spelt correctly.

Python is very precise. A missing colon, bracket or space can stop the program from running, so check one line at a time.

11 Download your code

Your environment detector is ready to transfer to the micro:bit.

1. Connect the micro:bit to your computer using the micro USB cable.
2. Select the **three dots** next to **Send to micro:bit**.
3. Follow the instructions on the screen.
4. Select **Connect** when asked.
5. Choose your micro:bit from the list.
6. Select **Send to micro:bit** to download the code.



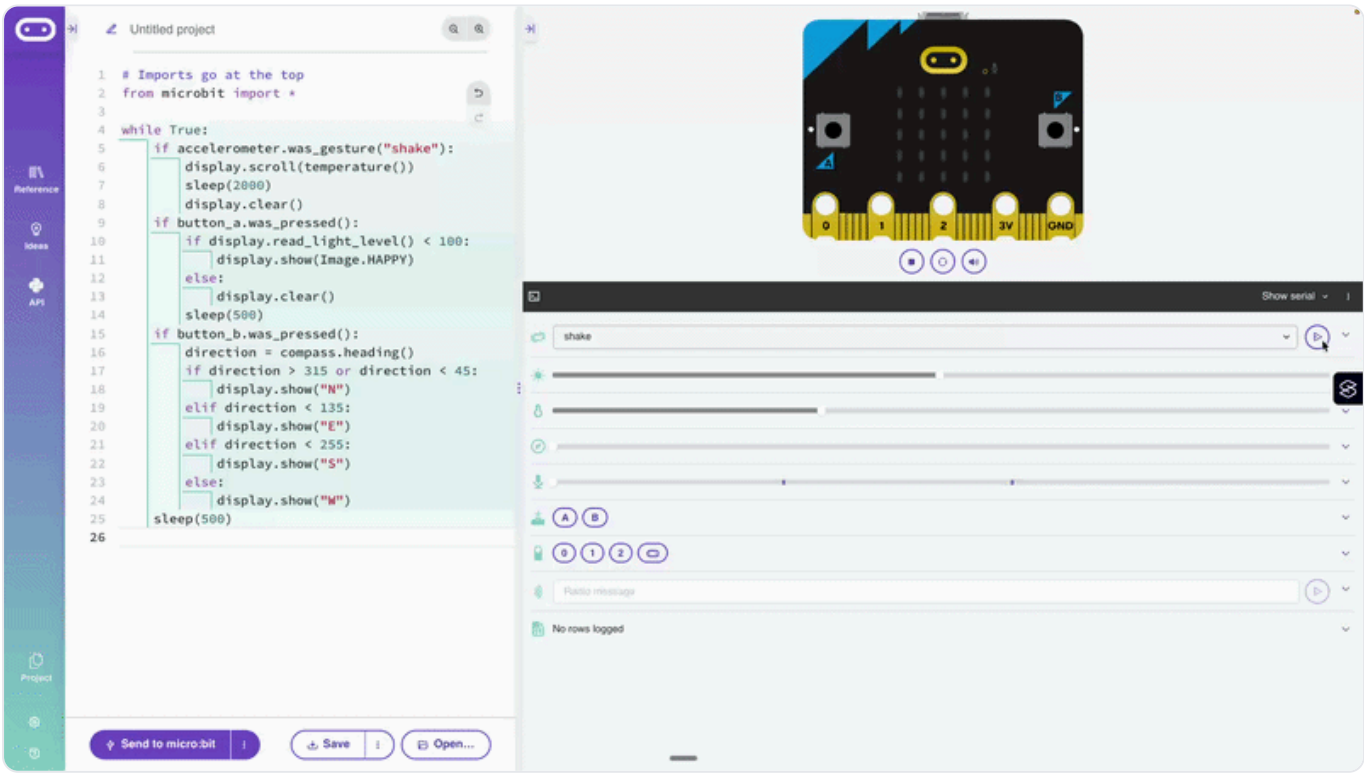
If the editor cannot find the micro:bit, try another USB cable. Some cables can only charge devices and cannot transfer code.

12 Test your environment detector

You can test the project using the simulator or a real micro:bit.

Test the simulator

The Python editor has a built-in micro:bit simulator.



Temperature

1. Find the shake control underneath the simulator.
2. Select the play button next to it.
3. Watch the temperature appear on the LED matrix.
4. Move the temperature slider and test it again.

Light

1. Find the sun symbol underneath the simulator.
2. Move its slider to change the light level.
3. Press button A on the simulator.
4. A happy face should appear when the light level is below `100`.

Compass

1. Find the compass control underneath the simulator.
2. Move the slider to change the direction.
3. Press button B.
4. Watch the direction letter appear on the LED matrix.

Test a real micro:bit

- **Temperature:** Shake the micro:bit to display the temperature.
- **Light:** Cover the micro:bit with your hand and press button A.
- **Compass:** Turn in a different direction and press button B.

> 💡 **Compass calibration**
>
> When the micro:bit starts, it may ask you to tilt it around until every LED is switched on.
>
> This is called calibrating. It helps the compass work out which direction is north.

13 Try it yourself

🚀 Try it yourself

Challenge: Build an environment warning system

Can you make the micro:bit warn you when the room gets too cold?

Inside the temperature section, add another `if` statement that checks whether `temperature()` is below a number you choose, such as `15`.

You could start with:

```
        if temperature() < 15:
            display.show(Image.SAD)
```

You could make the micro:bit show:

- `Image.SAD`
- `Image.SKULL`
- A picture of your own

Can you create another warning for the light sensor?

Try changing the button A code so the micro:bit shows a warning picture when the room gets too dark.

14 Stuck? Quick fixes

Problem	Try this
Nothing happens when I shake the micro:bit	Check that the line begins with `if` and ends with a colon: `if accelerometer.was_gesture("shake"):`. Make sure the display instructions are indented underneath it.
The temperature looks too warm	This is normal. The sensor is on the micro:bit itself, so it can warm up after the board has been switched on or held in your hand.
The happy face does not appear	Cover the micro:bit with your hand. Check that your comparison says `display.read_light_level() < 100` and that `display.show(Image.HAPPY)` is indented underneath it.
Button B shows the wrong direction	Complete the compass calibration by tilting the micro:bit until every LED has lit up.
The compass always shows W	Check the numbers in your comparisons: `315` and `45` for north, `135` for east and `225` for south.
I see an error message	Check your spelling, brackets, quotation marks, colons and indentation. Python needs these to be exact.
The code keeps showing a number before the letter	Check that you removed the first, easier button B section before adding the letter version.
My code will not download	Try another USB cable or USB socket. Check that the micro:bit is connected to the Python editor.

Don’t worry if it doesn’t work first time. That’s how coding works! Check one line at a time and compare it with the pictures.