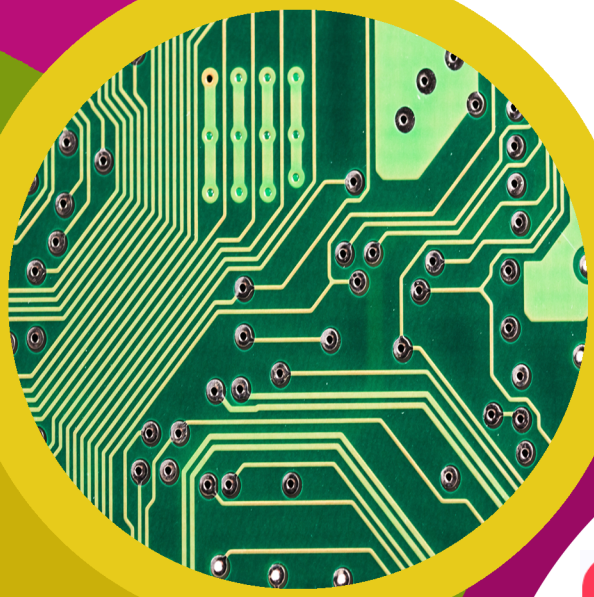
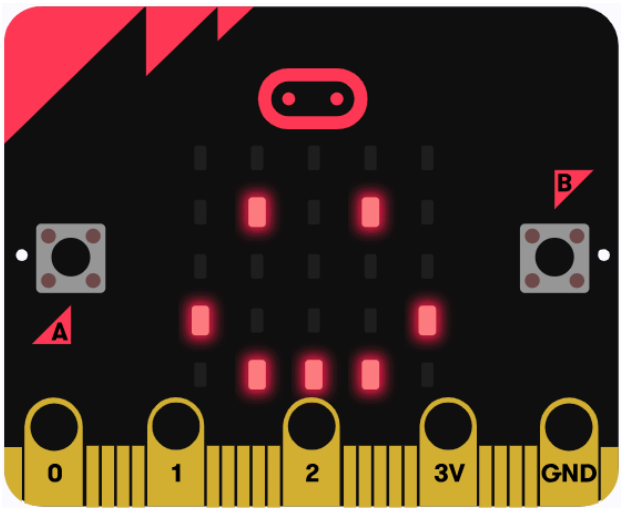
technocamps

Micro:bit Health and Wellness Teacher Guidance





UNDEB EWROPEAIDD EUROPEAN UNION
Llywodraeth Cymru Welsh Government
Cronfa Gymdeithasol Ewrop European Social Fund

Prifysgol Abertawe Swansea University
CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD
PRIFYSGOL BANGOR UNIVERSITY
Cardiff Metropolitan University | Prifysgol Metropolitan Caerdydd
University of South Wales Prifysgol De Cymru
i.t.wales
PRIFYSGOL ABERYSTWYTH UNIVERSITY
PRIFYSGOL glyndŵr Wrecsam | Wrexham glyndŵr UNIVERSITY

Links to Science and Technology AoLE

Computation:

(PS2) I can create simple algorithms and am beginning to explain errors.
(PS2) I can follow algorithms to determine their purpose and predict outcomes.

(PS3) I can identify repeating patterns and use loops to make my algorithms more concise.
(PS3) I can use conditional statements to add control and decision-making to algorithms.

Links to Other AoLEs

Health and Well-Being:

(PS2) I can describe the way in which physical and emotional changes are connected in different contexts.
(PS2) I can notice and communicate my feelings.

(PS3) I can self-regulate my emotions in a healthy way using strategies that I have developed.
(PS3) I can see the benefits of communicating about feelings as one of a range of strategies which can help promote positive mental health and emotional well-being.

The Four Purposes and Cross-Curricular Skills

This resource provides **Critical Thinking and Problem-Solving** opportunities. Students are required to follow and design an algorithm using block-based programming. They are able to analyse errors in the code, identify solutions, and deduce the next steps in the code.

Learners also use **Creativity and Innovation**. They are encouraged to discuss and implement strategies to improve their program. Learners are taught basic strategies to improve mental and physical health and how to program devices that aid in this. They are encouraged to create personalised messages and patterns when coding these devices.

The **Data and Computational Thinking** section of the **DCF** applies to this resource. Students will learn to code efficiently using loops, conditionals, and events to create several different algorithms that takes input from the user and the environment to display a useful output.

Why Is Learning This Important?

This resource provides learners with the opportunity to create simple algorithms with a demonstrable application, using a block-based programming language. It introduces concepts such as conditionals, loops, and events, which are critical to most common programming languages. This resource also teaches the importance of being active and healthy, focusing on mental, physical, as well as emotional health. This can be expanded to introduce students to text-based programming such as Python.

Suggested Approaches Key

In this suggested approach we use the following colours to differentiate the types of activities:

- **Yellow - Explain.** Teachers should explain the slide/example to the class.
- **Green - Discuss.** Teachers should start an open discussion with the class to get them to feedback some answers/ideas.
- **Purple - Activity.** Students are expected to complete an activity whether it be in their workbooks or on the computer, followed by a discussion of their solutions.
- **Green - Introduction/Conclusion.** The introduction/conclusion is also colour coded green. Teachers should hand out materials in the introduction and conclude the session and collect materials at the end.

Introduction

Begin with introductions, and a brief explanation of the Technocamps programme, before handing out any resources required by learners and any additional aids for learners with additional learning needs.

Explain: Topics Covered Today

We will be learning how to use micro:bits and block-based programming to create several devices that can help us with our mental and physical health.

Discuss: Micro:bit

Provide each student with a micro:bit and ask them what they know about it. Have they used them before? Can they tell you what some of the components are (e.g. buttons, LED lights, USB connector)?

Explain: What is micro:bit?

The micro:bit is a very small computer that is used to teach how hardware and software work together.
It has several components: 25 Led lights that can be used to display images, sensors that can detect light/temperature/movement, buttons, and radio and bluetooth antenna.
We can program the micro:bit to take input, display output, process information, communicate with other micro:bits and many more things.

Explain: What is programming?

Programming is telling a computer what to do using a set of ordered instructions. The set of ordered instructions is called an **algorithm**. The language used to tell the computer what to do is called a **programming language**.
Introduce the students to the MakeCode editor and explain how to connect

Explain: Downloading Programs

Each time you make changes to your program you need to click on the download button before the micro:bit can run the program.

Activity: Smile!

Program the micro:bit to display a happy face. Drag the **show icon** block into the **forever** loop and choose the icon from the dropdown menu.

Explain: LED lights

The **show icon** blocks allow us to display default images on the micro:bit. Using the **show leds** block, we can customise the image that is displayed.

Activity: Custom Icons

Ask the students to create their own icons using the **show leds** block. Clicking on each individual square turns that light on or off.

Emotion Badge

Explain: Emotion Badge

An emotion badge is a way to visualise how we are feeling using images rather than speech. For example we can show a smiling face when we are happy and a frowning face when we are sad.
This is useful if we are not comfortable talking about it or for non-verbal people.
We can program the micro:bit to be an emotion badge.

Explain: Events

Each micro:bit has two buttons: A on the left and B on the right.
These buttons allow us to choose which action to take without reprogramming the micro:bit each time.
For example, we can show a happy face when we press button A and a sad face when we press button B.
These commands are found in the **input** section.

Activity: Button Events

Create a program that displays one image when button A is pressed. Extend this program to display a different image when button B is pressed.

Explain: Additional Inputs

The micro:bit has several ways of taking input. These are found in the **Input** section. Some examples include shaking, tilting left or right, or pressing the logo.

Activity: More Emotions

The micro:bit also has many more icons that can be displayed. Ask the students to experiment with different inputs and show icon blocks to create a more complex emotion badge. Some examples are shown on slide 18.

Explain: Breathing Exercises

One method of reducing stress and anxiety is to use breathing exercises. This involves regulating your breaths (taking deep breaths in regular intervals) to help you relax.
Breathing along with an animated sequence can help us time our breaths. We can program the micro:bit to display an animated sequence that repeats over and over. To do this we need to use loops.

Explain: Loops

Loops can be used to repeat commands in a program without typing out each action every time. They can be repeated forever, for a certain number of times, or for a given condition.
In MakeCode, these are found in the **Loops** section.
For a loop to have a purpose, an action command needs to be placed inside it.
All MakeCode programs start with a default **forever** loop. This loop will run a set of commands until the micro:bit is unplugged or reset. You can only have one **forever** loop in a micro:bit code.

Activity: Changing Faces

Ask the students to create a program that displays two alternating images - a happy face and a sad face. They can do this in the default **forever** loop.
Suggest that they add a **pause** command between the two icons.

Breathing Exercise

Discuss: Forever vs. On Start

Explain the difference between the **on start** block and the **forever** loop. Ask the students what they think would happen if they put the code from the 'changing faces' activity in the **on start** block instead of the **forever** block.
Answer: The code would run once and then stop without repeating.

Activity: Breathing Exercise

So far we have used the default icons to create our animations. Ask the students to combine what they have learned about creating customised icons with the LED lights and using loops to create an animation.
This animation should help the user time their breathing. Depending on how many icons are involved in their animation, the students may need to add pauses throughout the loop to time it with their breathing.

Explain: Positivity Generator

A positivity generator is a device that displays a random positive statement when prompted.
It works in a similar way to a magic eight ball. A magic eight ball will show the user a random answer to a question when shaken.
We are going to create our own positivity generators with the micro:bit. To do this, we need to use variables.

Explain: Variables

Variables are stored values that are remembered by the micro:bit. We can use these variables and change them.
Variables can take different forms such as a number or a text string.
We can change a variable's value and use it many ways in our code. But first, we need to define it.
In MakeCode, they are found in the **Variable** section.

Activity: Creating Variables

Instruct the students on how to create variables.
Click on the **Variables** section and **Make a Variable**. Create one variable called quote.

There should now be a few more commands available for selection in the **Variables** section.

Activity: Positivity Generator

Instruct the students on how to make a positivity generator with the micro:bit. Go through the slides step-by-step, asking the students for ideas before moving on.

1. Start by adding the **on shake** input block. We want the positive message to be displayed when we shake the micro:bit.

2. Add a **set quote to 0** block from the **Variables** section to the input block.

3. Add a **pick random 0 to 10** command to the code. Change the numbers to **1 to 5**, assuming we will be creating a positivity generator with 5 different quotes.

When we shake the device now, the variable will be set as a random number between 1 and 5. However, the micro:bit does not do anything with this information yet. We need to program the micro:bit to display a different output depending on which number the variable is set to. To do this we need to use conditionals.

Explain: Logic and If-statements

Conditions help us perform different actions based on different conditions. These conditions should always result in Yes/No or True/False.

For example:
If my homework is done, **then** I can go outside to play.
If I have eaten my dinner, **then** I can have dessert.
Ask the students for more examples.

In MakeCode, the if-statements are found in the **Logic** section.

Activity: Positivity Generator Continued

Continue instructing the students on how to make a positivity generator with the micro:bit. Go through the slides step-by-step, asking the students for ideas before moving on.

4. Add an if-statement to the code and expand the block by clicking on the ⊕ symbol.
5. Add a 0 = 0 block to each statement in the if block.
6. Drag the variable to each if-statement as shown on the slides. Change the number in each statement from 1 to 4.
7. Add a different quote to each if-statement using the show string block.

Now we have a working positivity generator. Encourage the student to try it out.

Exercise Counter

Explain: Exercise Counter

An exercise counter is a device that can help us keep track of an amount of something.
We can use this counter to count anything exercise related such as star jumps, laps around a track, or goals scored in football.

We can create a counting device using the micro:bit.

Discuss: Coding an Exercise Counter

To program our micro:bit to act as an exercise counter, we want to display the number using one button, and increase the number using another button.

Discuss with the students about how you could go about making an exercise counter.
Hint: We need to use variable and events.

Activity: Exercise Counter

Ask the students to create an exercise counter with their micro:bit.

First they will need to create a new variable with a suitable name such as "count" or "number".
Drag the **set count to 0** block and a **show number** block to the **on start** block. Add the variable **count** to the **show number** block.

Before moving on, ask the students how we can add a command to increase the number.

Add two event blocks, one for button A and one for button B.
To the button A block, add a **change count by 1** command. To both the button A and button B blocks, add a **show number count** command.

Ask the students to test their code. The counters should now work.

Activity: Counter Reset

Ask the students if they have any ideas about how to add a reset feature to the counter.

Add another event block - **on button A + B pressed** - to the code.
Drag a **set count to 0** and a **show number count** block to the event block.

The counter should now reset if we press both buttons at the same time.

Step Counter

Explain: Step Counter

A step counter, also called a pedometer, senses the number of steps you take and keeps track of that number. Using the micro:bit, we can create a step counter. This will be very similar to the exercise counter already made, but we want the micro:bit to react to movement instead of pressing a button.

Ask the students if they know how we can use movement as an input.

Discuss: Movement Input

There are different ways we can code the micro:bit to sense movement. One option is to use the **on shake** block. In this case, the micro:bit will carry out an action when it is shaken.

Another option is to use an **if-statement** to make the micro:bit react to a certain amount of acceleration. This value can be changed so can be customised to each person's step.

Ask the students which step counter they think is better. Is one easier to code? Is one of them more sensitive? Can they be customised?

Activity: Step Counter

Ask the students to create a step counter using the micro:bit.

Start by creating a new variable called "steps". Set the steps variable to 0 and display this number **on start**.

Add an **if-statement** to the forever loop and change the condition in the statement to **if acceleration(mg) strength = 1500**.

Add a **change steps by 1** command and a **show number steps** command to the **if-statement**.

Add some commands to the button A and button B input blocks. On button A, show the current number of steps. On button B, reset the step counter. The code for this is shown on slide 50.

Activity: Using Your Step Counter

Get the students to run their code and see if it works.
The micro:bit can be attached to their shoe using an elastic band. Then they can try to walk around the room to test their step counter.

Does the step counter accurately count the steps? The acceleration value in the **if-statement** may need to be changed depending on each person's step.

Differentiating for Learners

- There are several ways to extend the activities provided to challenge students who are more confident with programming. For example, they can extend their step counter to save their movement data and plot it on a graph using the *datalogger* extension in MakeCode.
- Some learner may need more guidance in assembling the code than others. Since this workshop is aimed at ages 9-11, there might be several learners with little coding experience so will require step-by-step instructions in assembling the blocks. Other learners can use their previous coding experience to write the algorithm without any code provision.
- The micro:bit website and editor allow for navigation using accessibility

Where To Go Next

- Extensions can be added to many of the activities. For the emotion badge, the learners can use the micro:bit radio feature to send emotion icons to each other. For the step counter, they can log their movement data and plot the x-, y-, and z-coordinates of their movement.
- Micro:bit also has a user-friendly Python editor. This workshop can be adapted to introduce learners to a text-based programming language.
- There are many other devices that can be made with the micro:bit that promote physical and mental health, that may require more coding experience. For example, they can make a heart rate monitor, a stopwatch, or a simple game to reduce stress.

technocamps

@Technocamps

Find us on Facebook