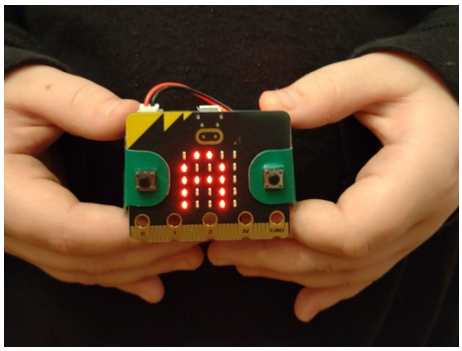
Lesson Plan

Lesson 1: Name badge

Ages: 7 – 11

Programming language: MakeCode blocks

Topics: Algorithms (Computational thinking), Input/output (computer systems), Iteration/loops (Programming)
Outcomes: Students learn how to code the micro:bit for the first time by making a name badge.



Key learning in this lesson

- Understand the micro:bit is a tiny computer which needs instructions in code to make it work.
- Understand that sets of instructions for computers in a sequence are also called algorithms or programs.
- Use the MakeCode editor to create instructions in code that the micro:bit can understand and then transfer them to the micro:bit.
- Know the micro:bit has an LED display output which it can use to show words (as well as numbers and pictures).

Learning objectives

- I can explain that the micro:bit is a tiny computer.
- I can explain that computers need to be given set of instructions (an algorithm) in code.
- I can give the micro:bit instructions in code to make a name badge using the LED display output.

Preparation: before the lesson

If you're new to the BBC micro:bit, our Get started guide has everything you need to get up and running: https://mbit.io/lessons-get-started

What you need

- BBC micro:bits and micro USB cables – at least one for every two students
- At least one computer (laptop or desktop) for every two students, with internet access to the Microsoft MakeCode editor: https://makecode.microbit.org/
  - Alternatively, you can use iPads with the micro:bit app installed. See our guide: https://mbit.io/lessons-mobile
- micro:bit battery packs (optional) – one per micro:bit
- PowerPoint presentation – whole class teaching slides
- Student handout (optional) – code blocks you can cut out and assemble, plus the completed project blocks with annotations explaining what each block does.
- The lesson download includes optional printable certificates and learning journeys for all 6 projects in the whole unit of work.

The lesson download also includes an optional '.hex' program file of the completed project, which may be useful if you have limited internet access. You can drag and drop this direct onto the MICROBIT drive when you connect a micro:bit to your computer. You can also drag '.hex' files into the MakeCode editor to examine the code and test it in the simulator.

Differentiation ideas

- If this is your first coding lesson, or first time using the micro:bit, it may be hard to know which students will need more support. You can use the extension ideas in the teaching section below for students who complete the task more quickly than others.

Decide how to deliver the 'Create' coding activity

You'll share the completed code on screen with your whole class from the slides. You can choose any of these methods that suit your classroom and teaching style:

- You (or selected students) model building and testing the code yourself on a large screen. The completed code is in the lesson plan and slide deck.
- Give students printed code blocks handouts to follow or cut out and assemble.
- Share a step-by-step YouTube coding video with the class, or individual students.
- If YouTube is blocked in your school, we also provide an animation in the slides showing how to assemble the code.
- Students can individually follow an online step-by-step tutorial.

- You can also choose to manage the whole class coding activity and save every student's code using **micro:bit classroom**. Find out more at https://mbit.io/lessons-classroom

Decide how to deliver the 'Evaluate' activity

Students download their code to real micro:bits and test the project.
You may want your students to answer the evaluation questions:

- on paper
- verbally with partners
- as part of a whole class discussion.

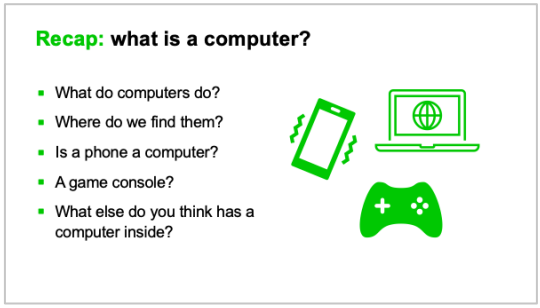
Glossary

algorithm:	a set of step-by-step instructions
code:	instructions written in a way a computer can understand
hardware:	a physical device like a computer or a micro:bit that is told what to do by computer programs (software)
LED:	light emitting diode - the micro:bit display is made of 25 LEDs
loops:	allow you to repeat sets of instructions without having to write them out multiple times
output:	data sent from a computer such as information shown on the LED display
program:	a set of instructions written in code that performs a given task
software:	a set of programs used to perform a task, which is usually more complex than a single program
string:	a collection of letters, numbers or symbols stored in a computer, for example the letters that make up your name

Teaching: during the lesson

Recap prior learning (slide 2)

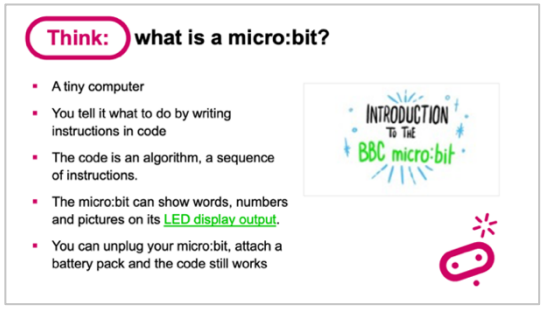
Ask your students what they think a computer is. See if they know that computers are not just laptops and desktops PCs, they all around us, inside phones, games consoles, TVs and even cars.



Think: starter activity

What is a micro:bit? (slide 3)

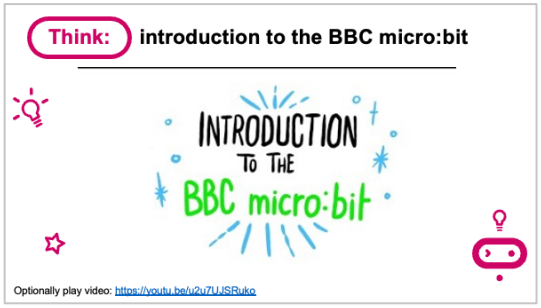
Share slide with your class.



- A tiny computer
- You tell it what to do by writing instructions in code.
- The code is an algorithm, a sequence of instructions.
- The micro:bit can show words (and numbers and pictures) on its LED display output.
  - **LEDs** are light emitting diodes, the lights on the front of the micro:bit. Information sent out of a computer is called an **output**.
- You can unplug your micro:bit, attach a battery pack and the code still works.

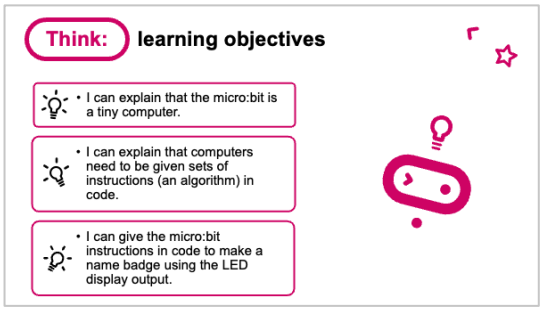
Introduction to the BBC micro:bit video (slide 4)

- Optionally play YouTube video: https://mbit.io/lessons-microbit-video



Learning objectives (slide 5)

- I can explain that that the micro:bit is a tiny computer.
- I can explain that computers need to be given sets of instructions (an algorithm) in code.
- I can give the micro:bit instructions in code to make a name badge using the LED display output.



Explain today we're going to code our micro:bits to show our names. Our names are shown scrolling across the LED lights on the front of the micro:bit. We'll create an algorithm – instructions in code, a language the micro:bit can understand.

Name badge introduction video (slide 6)

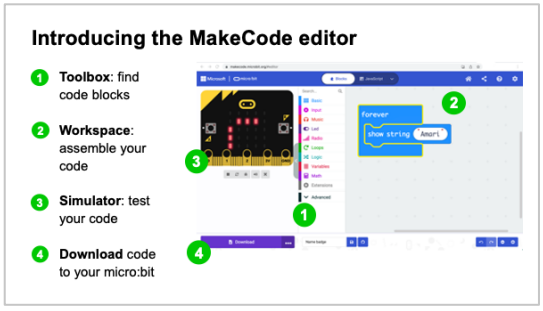
- Optionally play project introduction video: https://mbit.io/lessons-name-intro-video

Create: coding activity

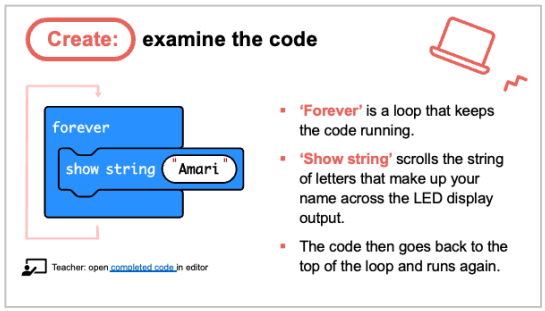
Introduce the parts of the MakeCode editor (slide 7)

If it's your first lesson with the micro:bit, introduce the parts of the MakeCode editor. The MakeCode editor works a lot like Scratch.

- 1 Find code blocks in the Toolbox.
- 2 Assemble code blocks in the Workspace.
- 3 test your code in the Simulator.
- 4 download code to your micro:bit.



Examine code with students (slide 8)

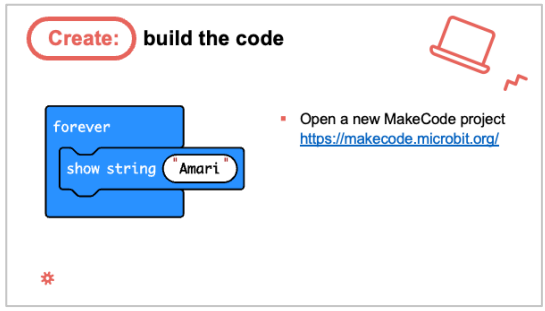


Explain:

- **'Forever**' is a loop that keeps the code running.
- '**Show string**' scrolls the string of letters that make up your name across the LED display output.
- The code then goes back to the top of the loop and runs again.

- You can also follow the link in slide 8 to open the completed code in the editor: https://mbit.io/lessons-name-code
  Model changing the name in the 'show string' block and seeing it change in the simulator.

Model building the code (slides 9-12)



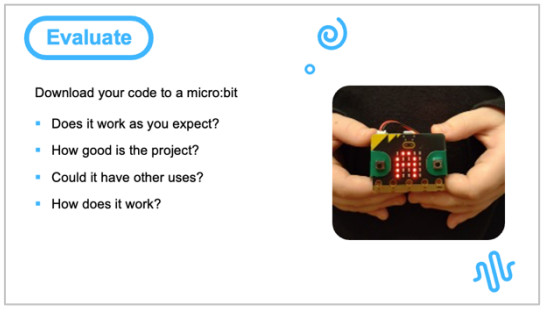
- You can open a new MakeCode project from slide 9 and model building the code. All the blocks you need are in the 'Basic' section.
- Optionally show the class the YouTube coding video on slide 10: https://mbit.io/lessons-name-code-video
- Or share the coding animation in slide 11 if YouTube is blocked in your school.

Students recreate the code in the MakeCode editor, using their own names, testing it in the simulator. They can:

- Copy the code in slide 9.
- Follow printed code blocks handouts.
- Individually follow a step-by-step online tutorial: https://mbit.io/tutorial-name-badge - you can share the link from slide 12
- If you are using micro:bit classroom, start a new session and ask your students to join your lesson. You can also open a session with completed code to edit and share with students: https://mbit.io/lessons-name-classroom

Evaluate: (slide 13)

Students transfer code to their micro:bit and test. If you have a battery packs, encourage students to unplug micro:bits from computers and attach batteries. Their code remains on the micro:bit and will still work.



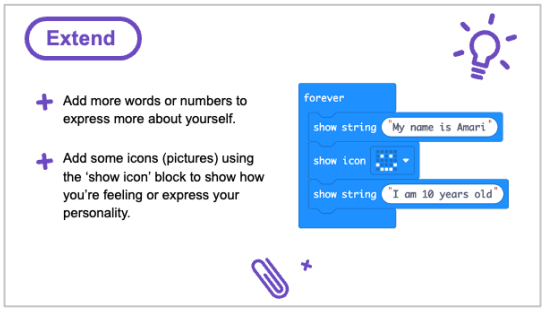
Questions:

- Does it work as you expect?
  - If not, do you need to debug the code and download it again?
- How good is the project?
  - Can you think of anyone who would like this project and find it useful of enjoyable?
  - How could you improve it?
- Could it have other uses?
- How does it work?
  - Encourage students to think about how it works when holding it in their hands.

Extend: (optional, slide 14)

If students finish early, they can remix their code to explore different uses of the LED display output:

- Add more words, or numbers to express more about yourself.
- Add icons (pictures) to show how you're feeling or express your personality.



Share: revisit learning objectives (slide 15)

- I can explain that the micro:bit is a tiny computer.
- I can explain that computers need to be given set of instructions (an algorithm) in code.

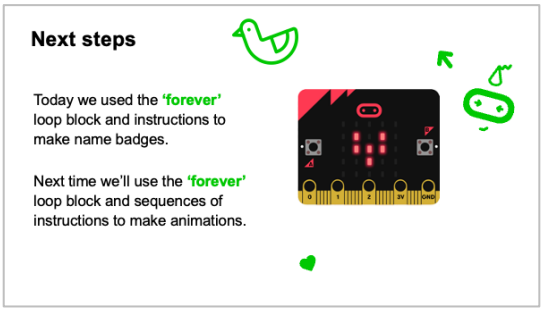
- I can give the micro:bit instructions in code to make a name badge using the LED display output.

Ask:

- How did you tell your micro:bit what to do? (e.g., used MakeCode blocks to make an algorithm or instructions in code that the micro:bit can understand)
- What code did you use? (‘forever’ loops, ‘show string’, possibly ‘show icon’ blocks)
- What parts of the micro:bit did you use? (LED display output)
- Did you change the code? (I used my own name; I added more words and pictures to say more about myself.)

Next steps (slide 16)

Today we used the ‘forever’ loop block and instructions to make name badges. Next time (in Lesson 2: beating heart) we’ll use the ‘forever’ loop block and sequences of instructions to make animations.



Assessment: after the lesson

When assessing students’ work you may find it helpful to ask these questions:

To what extent can they explain that that the micro:bit is a tiny computer, and understand that other devices like laptops, tablets, phones, and game consoles are also computers?

What is their understanding of computers as machines that need to be given sets of instructions (an algorithm) in code, a language computers can follow?

How successful was the student in giving the micro:bit instructions in code to make a name badge using the LED display output?

Here are some guiding criteria you might want to include when assessing your students' work:

WORKING TOWARDS the learning objective

The student may know what a computer is but does not recognise other devices like micro:bits, phones, or games consoles as computers.

The student knows what instructions are but cannot explain that algorithms are sets of instructions, nor that 'code' is a way of writing instructions so that computers can follow them.

The student created some code to display words or pictures in the MakeCode simulator, but not their own name and may not have transferred the code to a micro:bit.

MEETING the learning objective

The student can explain that the micro:bit is a computer, similar in some ways to a laptop, phone or games console.

The student can explain that computers, like the micro:bit, need to be given instructions (algorithms) written in code, a form computers can follow.

The student created code to show their name and transferred it to a micro:bit.

EXCEEDING the learning objective

The student can explain that micro:bits, laptops, phones, and other devices are all kinds of computers which use inputs and outputs to process information using instructions (programs) written in code.

The student can explain that the computers need to be given sets of instructions (algorithms) in the form of code and that algorithms can be expressed in different ways, such as natural language as well as languages that computers can understand like MakeCode, Scratch or Python.

The student created code to show their name and transferred it to a micro:bit. They also experimented by modifying their code to add more words or pictures to explain more about themselves.