technology

Program a micro:bit: follow precise instructions

Lesson at a glance

The class meets the micro:bit and programs it, step by step, on their own devices: first a number shown on the LED grid, then that block deleted again, then their own name shown when button A is pressed and a face shown when button B is pressed. Everything is tried in the simulator first and then sent to a real micro:bit if you have them. The idea being taught is that the device does exactly what it is told and nothing more, so a program that does the wrong thing is a set of instructions to look at rather than a broken machine.

Learning objectives

- Program a micro:bit to show an output on a button-press input
- Frame the program as input, process and output and debug errors as they arise
- Explain that a micro:bit runs exactly the algorithm it is given

Before the bell — prep

One micro:bit per pair if you have them, otherwise the simulator at makecode.microbit.org does every step except the last. Build the program yourself once beforehand and connect one micro:bit, because the connect-and-download step is the one that eats time. Have the cables out and counted before the lesson rather than during it.

Materials

Item	Qty	Per	Source	Low-cost substitute
micro:bit (with USB lead)	1	pair	school devices	run the lesson on the IWB using the MakeCode simulator

Safety watch-point

Normal care with devices and cables. Supervise plugging and unplugging micro:bits so the connectors are not pulled sideways, and keep cables off the floor between desks. Nothing else on this lesson.

Lesson 27

Teaching moves

- **What we're building:** Hold up a micro:bit, or show the simulator, and say it is a real computer that does exactly what it is told. Ask what they think it could show on those lights. Do not open the editor yet.
- **The Project Editor:** Walk the screen on the board before anyone touches a block: the simulator on the left, the toolbox in the middle, the workspace on the right. Two minutes here saves ten minutes of pupils hunting for a block that is in front of them.
- **Add Our First Code!:** Run it in the simulator immediately. The number scrolling across the LEDs is the first thing they have made the device do, and it is worth stopping on.
- **Delete Some Code:** Do not skip this, and do not let them just drag the block aside. Deleting code is part of programming, and knowing you can take a block out without breaking everything is what makes a pupil willing to experiment later.
- **Display Your Name:** Name the two halves as you model it: pressing button A is what goes in, the name on the LEDs is what comes out, and the block in between is what the micro:bit does with it. Have them test it in the simulator by clicking the button on screen.
- **Show a Happy Face:** Ask before you build it what would happen if both blocks used button A. Try it once on the board, so they see that the device follows both instructions rather than choosing between them. Then set it to B and test both buttons.
- **Connect your Microbit:** Allow more time than you expect. Pair up pupils around whichever devices connect, and let anyone still in the simulator keep working, they are not missing the learning. If cables are short, bring the micro:bits to two or three machines rather than moving everyone.
- **Code talk:** Talk only, nothing typed and nothing saved. Ask what the micro:bit did that they did not expect, and what the instruction actually said when they looked at it. Draw out that it did what it was told, which is the whole idea of the hour.

What it should show

Most pupils finish with a micro:bit that shows their name on button A and a face on button B, working in the simulator, and on a real device if you have them. A pupil who has both buttons doing something different has met the objectives. The download step is the most likely to be unfinished and the least important to finish.

Misconceptions & interventions

- **Pupils think the micro:bit is broken when it does something unexpected.** — Turn it back to the instructions. Read the blocks aloud together in order and ask what the device was actually told to do. It followed that exactly, which is the point.
- **Pupils expect the name to show without pressing anything, because the earlier number appeared on its own.** — Compare the two starts on the board. One runs when the program starts, the other waits for a button. That difference is what the button block is for.
- **Pupils drag a block out of the workspace and think they have deleted it, so it keeps running.** — Show that a block sitting loose in the workspace is still there. Deleting means dragging it back into the toolbox, which is exactly what the delete step practised.
- **Pupils believe nothing counts until it is on a real micro:bit.** — The simulator runs the same program. Say so plainly, so the pairs who cannot connect do not think their work is worth less.

Differentiation

Emerging	Developing	Proficient
• Aim for one button doing one thing. That is the whole idea, and a second button adds nothing new. • Let them stay in the simulator; every step except the download works there.	• Both buttons, each doing something different. Ask them to say what goes in and what comes out for each one. • When something misbehaves, have them read their own blocks aloud before changing anything.	• Try a third thing on shake, and predict what will happen before running it. • Ask what the micro:bit would do if two blocks told it to show different things on the same button, then test the prediction.

Cross-curricular hook

English, through precise instructions. Reading your own blocks back aloud in order is the same skill as checking a set of written instructions makes sense to somebody who will follow them exactly.