technology

Scratch: loops, events and finding the bug

Lesson at a glance

Frame the class as programmers building Frog Frenzy: a frog that pops up at random for pupils to click and score. Model each step on the IWB while pairs code in Scratch — hide on green flag, a forever loop with random go-to and show/hide waits, a score variable, and a when-clicked event that adds a point and sound. Pairs run after every new block to catch bugs early. Close with a display-only code talk on a bug they fixed and one change they would make.

Learning objectives

- Add a loop and an event to a Scratch program so a sprite responds as planned
- Find a bug when the program goes wrong and fix it
- Explain in their own words what the loop or event does

Before the bell — prep

Have every device open on a blank Scratch project before the bell, plus one blank project on the IWB for modelling. Check speakers or headphones work for the pop sound. Pair at shared devices if kit is short. No physical kit beyond devices.

Materials

Item	Qty	Per	Source	Low-cost substitute
tablet or laptop with Scratch	1	pair	school devices	run the lesson on the IWB with pupils taking turns

Safety watch-point

Manage device cables and posture; keep volume moderate on shared headphones/speakers. No other physical hazards.

Lesson 29

Teaching moves

- **What we're building:** Set the scene at the IWB: they are the programmers today. Explain the watch-then-do routine — you model, they build, they run. Ask what games they know that keep score, and say bugs are normal and expected.
- **Create a New Project:** Model a new project and deleting the cat on the board. Circulate until every pair has a blank stage before moving on; stop anyone still editing the cat.
- **Add the Forest Backdrop:** Show the backdrop library button, not the sprite button. Name the slip aloud if anyone adds Forest as a sprite and fix it together.
- **Add the Frog Sprite:** Model adding the Frog. Before any code, ask 'Which sprite is highlighted right now?' — blocks on the wrong sprite is the most common early bug.
- **Make the Frog Disappear:** Build hide-on-green-flag on the IWB, then have pairs run it. Name the green flag as an event. Watch for pairs who never click the flag and think nothing happened.
- **Random Frog Appearance:** Model the forever loop slowly with the board zoomed in so blocks sit inside the loop's mouth, not under it. Point to each block as you place it. Ask 'Why does the frog keep reappearing?'
- **Hide the Frog again:** Add wait-then-hide inside the loop. Ask pairs to predict the new rhythm before they run. Check again that the new blocks landed inside forever, not outside.
- **Create a Score Variable:** Model creating the variable named score first, then the set-to-0 block. Ask what the scoreboard will count and point out the score now showing on the stage.
- **Score a Point:** Model when-this-sprite-clicked. Stress it only runs on a frog click, not automatically. Sound on for the pop. Watch for pairs clicking the stage instead of the frog.
- **Clear the Effect:** Place clear-graphic-effects at the top inside the forever loop. If the frog stays pixelated, the block is in the wrong place — fix that, then let pairs play the finished game.
- **Code talk — what worked and what we'd change:** Devices closed or turned away — talking is the work. Invite one pair who hit a bug to describe how they spotted it and what they changed; revoice their method so the class hears the process, not just the fix.

What it should show

A working game: green flag hides the frog; the forever loop waits, jumps to a random spot, shows for about 2 seconds, then hides again; clicking the visible frog plays a pop, bumps score by 1, and briefly pixelates. If the frog never reappears, blocks are usually outside the forever loop. If score never rises, the click script is on the wrong sprite or they are clicking the stage.

Misconceptions & interventions

- **Pupils drop wait, go-to and show under the forever block instead of inside its mouth, so the loop runs empty and the frog never reappears.** — Zoom the IWB on your forever block and physically point inside the C-shape. Have them drag the stack until it snaps in, then re-run the green flag.
- **Pupils think the score will go up by itself once the variable exists, without a when-clicked event.** — Hold up the two hats: green flag vs when-this-sprite-clicked. Ask which one only fires when they actually click the frog, then have them click and watch score change.
- **Pupils try to use set score to 0 before creating the variable, so the block is missing or greyed out.** — Stop and model Make a Variable named score first; only then pull set score to 0. Point to the score readout appearing on the stage as proof it exists.

Differentiation

Emerging	Developing	Proficient
• Work at the teacher table with a partly-built forever loop already snapped; pupil adds only the show/hide waits and runs the green flag. • Partner them with a peer who narrates each block name before it is placed.	• After a working game, change the wait times and re-run to feel how difficulty shifts. • Prompt: say in one sentence what the forever loop is doing that the green-flag hide is not.	• Fair-test style critique: what would break scoring if the click script sat on the stage instead of the frog? • Design one improvement using only blocks already in the project (faster waits, second variable, or an extra sound) and explain it in the code talk.

Cross-curricular hook

Link to Maths — the score variable is a live counter; pupils read how it changes by one with each successful click.