technology

Scratch: loops and events

Lesson at a glance

The class builds Penalty Shootout in Scratch, step by step, on their own devices: click a spot in the goal to take the shot, the ball travels there and shrinks as it goes, and a goalkeeper dives to a random spot to try to save it. The two ideas being learnt are the loop that carries the ball, and the events that decide when each script starts. Model each piece on the board, let them build it and run it, and close by talking about what they had to fix.

Learning objectives

- Use a loop to repeat instructions in a Scratch program
- Use an event, the green flag, a click on the goal, or a message being received, to control when a script starts
- Debug a program that uses loops or events and record a fix or a change you tried

Before the bell — prep

One tablet or laptop with Scratch per pupil or pair, and the board linked to your device. Build the game yourself once beforehand, and check two things while you do: that your devices can download the goal picture and upload it as a backdrop, and what colour the goalposts actually are in it. If uploads are blocked on school devices, the step offers a ready-made starter project with the backdrop already in place, so have that link open before the class starts rather than discovering it mid-lesson.

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

Normal care with devices, and supervise packing away at the end. Nothing else on this lesson, the work is all on screen.

Lesson 27

Teaching moves

- **What we're building:** Take two or three penalties on the board, missing at least one, then close it. Ask what starts the shot and what makes the ball travel. Collect their words before you supply loop and event. Keep the block palette closed until you model.
- **Download and add the goal backdrop:** Do this one together at the same pace, it is the step most likely to strand a group. If a device will not upload the picture, send that pair straight to the starter project link rather than letting them fall behind on a file problem that has nothing to do with the coding.
- **Choose your shot:** Be explicit that this code goes on the stage, not on the ball, and that a shot is taken by clicking the goal rather than the ball. Warn them that running it now appears to do nothing, and that this is correct: the message is being sent, but nothing is listening for it until the next step.
- **Move the ball:** Name the loop here. The repeat is what carries the ball across in thirty small steps instead of teleporting it, and the shrinking inside the same repeat is what makes it look like it is travelling away. Ask what would happen with the move block outside the repeat, then try it once on the board so they see the jump.
- **Make the goalkeeper dive:** Point out that the keeper and the ball both start from the same message. One click, two scripts running at once, which is the clearest example of an event they will meet today. The random direction is what makes the game worth playing, so expect saves to feel unfair, that is the point.
- **Detect if it hits the post:** The colour in this block has to be picked from the goalposts in your own backdrop with the colour picker, not typed or assumed. A pair whose post detection never fires has almost always got a colour that is close to the post but not on it. Zoom in on the board once and pick it together.
- **Score a goal:** Ask why the goal message can simply sit at the end without a check of its own. Draw out that the save and the post both stop the script early, so anything that reaches the last line got past both. That is worth naming: the order of the checks is what makes the last one correct.
- **Code talk:** Close with talk only, devices left as they are. Take two or three pairs on what did not work first time and how they tracked it down. Revoice loop and event from their own examples, then ask one pupil to say the difference between the two in their own words.

What it should show

Most pupils finish with a ball that flies to the clicked spot and shrinks, a keeper that dives randomly, and messages for a save, a post and a goal. A pupil with the ball moving on the message and the keeper diving has met the loop and event objectives. The post check is the one most likely to be unfinished, because it depends on picking the right colour, and that is a reasonable place to stop.

Misconceptions & interventions

- **Pupils think nothing happened when they run the broadcast step, and undo it.** — Say in advance that sending a message looks like nothing until something listens for it. Run the two steps together on the board so they see the pair working as one action.
- **Pupils believe the green flag is the only way a script can start, so the message and the stage click do not feel like real starts.** — Put the three hat blocks side by side on the board. One sets the game up, one fires when the goal is clicked, and two more fire when the message arrives. Run each alone so they see separate starts.
- **Pupils click the ball to shoot, because the ball is the thing that moves.** — Show that the code was added to the stage, so the stage is what listens. Clicking where you want the ball to go is also how you aim, which is why it is the backdrop and not the ball.
- **Pupils assume the goalposts are white because the block shows white, so the post check never fires.** — Use the colour picker on the actual post in your backdrop. Have the pair prove it by aiming deliberately at the post and watching for the message.

Differentiation

Emerging	Developing	Proficient
• Aim for the ball moving to the clicked spot, which is the loop and the event together. Sit these pupils where they can see the board stacks and let them run the program after every block. • If the backdrop upload stalls, put them on the starter project immediately so the coding time is not lost to a file problem.	• Build through the save detection. Ask them to point at the loop and at each event and say what starts it, so the words attach to blocks they have placed themselves. • When something misbehaves, have them say what they expected before they change anything.	• Add the post check and the goal message, then change one number and predict its effect first: how fast the ball travels, how far the keeper dives, or how much it shrinks. • Ask how they would let the player take a second penalty without clicking the green flag again, using only blocks already on screen.

Cross-curricular hook

PE and English oral language. The class already knows what is fair in a penalty, so ask whether a keeper who dives at random is fair, and have them argue it in sequence: what the code does now, what they would change, and why that would be better.