LESSON 23

# Maze Game

60 minutes | Dynamic Digital Projects

## WHAT THIS LESSON DOES

This lesson guides students through creating a Maze Game using Scratch. They'll learn to program a character to navigate a maze, modify sprite size and position, and code keyboard controls. The lesson also includes instructions on how to prevent the character from passing through maze walls.

## WHAT STUDENTS SHOULD BE ABLE TO DO

- Understand and apply basic programming concepts using Scratch.
- Manipulate sprite properties such as size and position.
- Program keyboard inputs to control sprite movement.
- Implement collision detection to prevent sprite from passing through obstacles.
- Develop problem-solving skills and creativity by modifying and enhancing the game.

## BEFORE THE LESSON

- Open the Scratch starter project yourself and check it loads on the school network and past any content filter.
- One device per pupil, or per pair, with Scratch reachable in the browser.
- Have the block sequence for each arrow key to hand so you can model the point-and-move pattern on the board.

# Maze Game

### HOW THE LESSON RUNS

| |
|---|
| Introduction to Maze Game |
| Open the starter project |
| Add the Beetle sprite |
| Shrink the beetle |
| Program the arrow keys to move the beetle |
| Stop the beetle going through the walls |
| Wrap up and Play! |

### TEACHING MOVES

- 2. Open the starter project. Have everyone open the same starter link and confirm the maze sprite is showing before anyone starts coding. Wait for the slowest device to load.
- 3. Add the Beetle sprite. Model finding Beetle in the sprite library once on the board. Point out the beetle, not the maze, must be selected before adding any code.
- 4. Shrink the beetle. Get pupils to click the green flag after adding the size and go-to blocks. If the beetle does not shrink or jump to the start, the blocks are on the wrong sprite.
- 5. Program the arrow keys to move the beetle. Build one arrow key together, then let them copy the pattern for the other three. Stress the four directions: up 0, down 180, left -90, right 90.
- 6. Stop the beetle going through the walls. Ask why we move -10 steps. Draw out that it undoes the 10 steps forward, so the beetle bounces off the wall instead of passing through.
- 7. Wrap up and Play! Give a few minutes of play, then challenge finishers to add an obstacle or a second level rather than just replaying.

### WHAT TO WATCH FOR

- Pupils add code while the maze sprite is selected, so the arrow keys and shrink blocks do nothing to the beetle. Show them the highlighted sprite at the bottom of the screen. The blocks only act on whichever sprite is selected, so click Beetle first.
- Pupils think the beetle should stop dead at a wall, and are confused when the game seems to jitter instead. Explain the beetle is not stopping, it is being pushed back the same distance it just moved. That is what keeps it out of the wall.
- Pupils mix up the direction numbers and the beetle moves the wrong way for a key. Put the four values on the board: up 0, down 180, left -90, right 90. Have them test each key one at a time.

**DIFFERENTIATION**

- Emerging: Pre-open the starter project on their device so they begin at the coding, not the loading. Give the direction numbers for the arrow keys as a printed strip beside the device.
- Developing: Let them copy your one worked arrow key and adapt it, then test each key as they go rather than building all four first.
- Proficient: Once collision works, challenge them to add a second obstacle sprite or a finish line that ends the game when touched. Ask them to explain their touching-then-move-back logic back to you in their own words.