LESSON 6

Space Shooter

60 minutes | Coding and Interactive Creators

WHAT THIS LESSON DOES

Embark on an exciting journey to create your own 'Space Shooter' game. Start by setting up a new Scratch project, add a starry backdrop and create your game variables. Add a rocketship sprite and program it to follow your mouse pointer. Introduce a ball sprite for the rocketship to shoot and a balloon sprite for targets. Implement scoring and lives system, and finally, set up the game over conditions. Enjoy playing your own game and think about how you can improve it.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Develop skills in creating and managing a new Scratch project.
- Understand and apply the concept of variables in game development.
- Gain proficiency in adding and programming sprites in Scratch.
- Learn to implement game mechanics such as scoring and lives.
- Enhance problem-solving skills by improving and modifying the game.

BEFORE THE LESSON

- Open scratch.mit.edu on the projector and check the school filter is not blocking it. If pupils are saving work, make sure they are logged into their Scratch accounts.
- One device per pupil or per pair. This is a long build, so pairing is a fair fallback if devices are short.
- Sound on and headphones out, as the game uses a pop sound effect when balloons are hit.
- Have the completed project open on your own device so you can show the finished game and check code against it.

Space Shooter

HOW THE LESSON RUNS

- Create a new Scratch project
- Add the Stars backdrop
- Create 'score' and 'lives' variables
- Setup the variables
- Add the Rocketship sprite
- Program your Rocketship
- Add the Ball sprite
- Add the Balloon1 sprite
- Detect a mouse click
- Fire a ball
- Make the balloons appear
- Score a point
- Loose a life
- Game over
- Play the game!

TEACHING MOVES

- 1. Create a new Scratch project. Show that every new project starts with the cat, and deleting an unwanted sprite is a normal first move. Right-click the cat and delete it together before anyone starts.
- 2. Add the Stars backdrop. Point out the category list on the left of the backdrop library so pupils filter rather than scroll endlessly to find Stars.
- 3. Create 'score' and 'lives' variables. Say a variable is just a labelled box that holds one number. Name the two boxes on the board before pupils make them, so the names match your later code exactly.
- 4. Setup the variables. Stress that setting score to 0 and lives to 3 must sit under the green flag, so the game resets every time it starts rather than carrying over old numbers.
- 5. Add the Rocketship sprite. Model the Costumes tab and rotating the rocketship to point right. This visual step trips pupils up, so walk it slowly and check screens.
- 6. Program your Rocketship. Explain the if-then condition out loud: only if the up arrow is pressed does the rocket move. Have pupils test that the ship points at the mouse before adding movement.
- 7. Add the Ball sprite. Point out the ball is hidden at the start on purpose. It will appear only when fired, so a blank screen here is correct, not a mistake.
- 8. Add the Balloon1 sprite. Same pattern as the ball: shrink and hide. Ask pupils why we hide it now, linking to the clones they will make appear later.
- 9. Detect a mouse click. Have everyone click the Stage box until it shows a blue border before coding. The when-stage-clicked block only works here, and pupils commonly drop it on a sprite by mistake.
- 10. Fire a ball. Explain broadcast as sending a message the ball listens for. Connect it back to message1 from the stage so pupils see the two halves working together.
- 11. Make the balloons appear. Warn pupils to add the new code under the hide block, not over it. Show that pick random creates a new balloon every one to five seconds.
- 12. Score a point. Point out the new code goes under the if-on-edge-bounce block. Have pupils test that hitting a balloon pops it and score climbs before moving on.
- 13. Loose a life. Show the second if-then handles the balloon touching the rocket. Note there are now two if blocks doing different jobs, so placement matters.
- 14. Game over. Explain the forever loop keeps checking whether lives has reached 0, and stop-all ends everything. Test by deliberately losing three lives together.
- 15. Play the game! Give real play time first, then pose the improvement ideas as a challenge. Ask pupils which blocks they would add or change for a game over backdrop or explosion.

WHAT TO WATCH FOR

- Pupils replace or overwrite existing code when a step says to add new code underneath, breaking the sprite's earlier behaviour. Point to the exact comment lines in each step and model dragging new blocks below the existing stack rather than starting fresh.
- Pupils try to add the when-stage-clicked block to a sprite and cannot find it or it does nothing. Remind them that block only lives on the Stage. Have them click the Stage box and confirm the blue border before looking for it.
- Pupils expect the ball and balloons to be visible from the start and think their code is wrong when the screen looks empty. Explain the hide blocks are deliberate and the sprites appear only when fired or cloned. Ask them to click the green flag and wait for a balloon to pop up.

DIFFERENTIATION

- Emerging: Pair them with a partner and let them mirror the build step by step. Check screens after steps 5 and 9, the two most common sticking points.
- Developing: Have them explain back to you what one block stack does, such as why lives changes by -1, before moving to the next step.
- Proficient: Point them at the improvement ideas in step 15: a game over backdrop, a lose-a-life sound effect, or an explosion costume for the balloons.