LESSON 10

Space Dodge

60 minutes | Game Design Studio

WHAT THIS LESSON DOES

Get ready to build an exciting game with MakeCode Arcade! You'll create a spaceship to dodge incoming asteroids. Follow step-by-step instructions to design sprites, control movements, and set game rules for a thrilling space adventure.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Develop skills in using MakeCode Arcade to create a game.
- Understand and apply the concept of sprites in game development.
- Implement controls for game characters using code.
- Apply the concept of randomisation in game elements for varied gameplay.
- Understand and implement game mechanics such as collision detection and life count.

BEFORE THE LESSON

- Open arcade.makecode.com yourself first and confirm the school filter is not blocking it. Sign in or check pupils can create a new project.
- One device per pupil, or per pair if devices are short. This is a long build, so charged devices matter.
- Have your own working Space Dodge finished and saved so you can show the target and check any stuck pupil against it.

Space Dodge

HOW THE LESSON RUNS

Introduction to Space Dodge
Create a new Arcade project
Create your spaceship sprite
Control the spaceship
Set the number of lives
Create the asteroids
Set their position
Set their velocity
Auto destroy
Lose a life
Game On!

TEACHING MOVES

- 1. Introduction to Space Dodge. Show your finished game running for ten seconds. Let them see the spaceship dodge and lose a life, then say they are building exactly this today, block by block.
- 2. Create a new Arcade project. Walk the whole room through reaching a blank project together before anyone codes. Fix logins and site access now, not mid-build.
- 3. Create your spaceship sprite. Point out the ship must face right because asteroids come from the right. Let them design freely in the sprite editor but insist they rename the sprite to spaceship.
- 4. Control the spaceship. Have them test the arrow keys immediately after adding this block. Check the ship cannot slide off screen before moving on.
- 5. Set the number of lives. Confirm the life counter appears on screen. If it does not show, the lives block is in the wrong place.
- 6. Create the asteroids. Explain the one-second interval spawns a new asteroid repeatedly. Get them to run it and watch asteroids appear before adding position or movement.
- 7. Set their position. Draw out that random up-or-down placement is what makes each play different. Ask them to run it and confirm asteroids start on the right at varied heights.
- 8. Set their velocity. Now the asteroids should move left across the screen. Have them test and adjust nothing else until movement works.
- 9. Auto destroy. Explain in plain terms why this matters: without it the game remembers every asteroid forever and slows down. Frame it as tidying up, not decoration.
- 10. Lose a life. This is the fiddly one. Point at the two dropdowns, Player and Enemy, and stress they must use otherSprite in the destroy block, not mySprite. Then let one asteroid hit them to test.
- 11. Game On! Give them time to actually play. Ask a few to explain one block that makes the game work, spoken not written.

WHAT TO WATCH FOR

- Pupils use mySprite instead of otherSprite in the destroy block, so the wrong sprite is destroyed or nothing happens on collision. In step 10, stop the class and show the difference on screen. mySprite is the ship, otherSprite is the asteroid that hit it. Only the asteroid should be destroyed.
- Pupils select the wrong kinds in the overlap block, so collisions are never detected and lives never drop. Check both dropdowns read Player and Enemy. A game where hits do nothing almost always means these are wrong.
- Pupils think the game runs slowly because their device is weak, not because sprites are piling up. Use step 9 to name the real cause. Show that removing the auto-destroy block makes it lag, proving old asteroids are the problem.

DIFFERENTIATION

- Emerging: Pair them with a partner for the login and first project so they are not left behind at step 2. Let them copy each code block exactly and test after every step rather than reading ahead.
- Developing: Prompt them to run and test after each block instead of building several at once, so errors are easy to find. Ask them to point to where each block lives and say what it does before adding the next one.
- Proficient: Once their game works, invite them to change the spawn interval or asteroid velocity and describe what the game does differently. Have them redesign the spaceship sprite or add a second life-losing effect, then explain their change to a peer.