technology

Finish, test and share

Lesson at a glance

This is the finishing sitting for Galaxy Ghosts in MakeCode Arcade. Pairs reopen their saved project and complete the last coded pieces — enemy speed variable, projectile firing on button A, projectile-enemy overlap for scoring, the Status Bar health extension, damage and game-over. They run and test the game against their original design brief, find and fix at least one bug, then share the build and explain one change. The lesson closes with a code talk on debugging.

Learning objectives

- Finish the game and test it against the original Design Brief plan
- Share the game and explain one design or code change made during the build
- Reflect on the process of creating and debugging the program, not only the product

Before the bell — prep

Have devices charged and the MakeCode Arcade site (arcade.makecode.com) bookmarked. Check each pair's saved project actually opens before the bell — have a fallback pairing ready for any missing file. Keep the original design brief for each pair to hand so they can test the finished game against their plan. Cue the IWB to model the overlap block, the usual bug spot.

Materials

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

Lesson 25

Teaching moves

- **Start a New Project:** Have pairs reopen their saved project, not start fresh — check names on screen as you circulate. If a file is lost, pair them with a group that has theirs so no one is stranded without a working build.
- **Create a speed variable:** Model creating the 'speed' variable on the IWB — spelling and picking the right variable both matter. Ask 'what does this variable store?' Watch for enemies that never fall because the velocity was not set to the variable.
- **Fire!:** Model choosing the projectile sprite and pressing A in the simulator. Watch for pairs whose projectile appears but sits still, or who skipped choosing an image and get a blank projectile.
- **Hit the target:** Model the overlap block on the board — this is the biggest bug spot. Stress selecting kind Projectile and kind Enemy, and using otherSprite (not mySprite) in the destroy block. Run it and confirm the score rises on a hit.
- **Add the 'status bar' extension:** Model adding the Status Bar extension FIRST — the health blocks only appear afterwards. Watch for pairs hunting for health blocks before the extension is in.
- **Take damage:** Model the enemy-player overlap. Remind pupils to select kind Enemy and kind Player and to use 'sprite' in the destroy block. Have them test by letting a ghost crash in and checking health drops by 10.
- **Wrap Up:** This is the testing moment: ask each pair 'does your game do what your plan said?' Have them compare the finished game to their design brief and note anything that differs — that difference is the change they will share.
- **Code talk — what worked and what we'd change:** Draw out a real bug a pair hit and how they tracked it down, then revoice their thinking so the whole class hears the debugging process. Invite a few pairs to name one change they would make next time.

What it should show

A working game: the ship moves left/right, ghosts fall from random positions and speed up over time, pressing A fires a projectile, a hit destroys the ghost and adds 1 to the score, and a ghost touching the ship drops health by 10 until game over at zero. A game where hits don't score usually has mySprite instead of otherSprite in the overlap block, or the wrong kinds selected — re-check that block first.

Misconceptions & interventions

- **Pupils think using 'mySprite' in the overlap block is the same as using 'otherSprite'.** — Open the onOverlap block on the IWB and point to sprite versus otherSprite — the projectile is 'sprite', the ghost it hit is 'otherSprite'. Swap it and run; the score now rises when a ghost is hit.
- **Pupils try to add the health blocks before adding the Status Bar extension and can't find them.** — Show that the special status bar blocks only appear in the toolbox after the extension is loaded. Add the extension together, then point to where the new blocks show up.
- **Pupils think 'finished' means the game just runs, not that it matches what they planned.** — Put their design brief beside the running game and go point by point — does it do what the plan said? Any mismatch is a real design decision to name and share, not a failure.

Differentiation

Emerging	Developing	Proficient
• Have this pupil focus on confirming one piece works — run the game after each block and check just that the ship moves or a ghost falls, before moving on. • Pair with a group at the teacher table for the overlap block, the trickiest step.	• Ask them to change the starting speed value and predict how the game will feel, then test it. • Prompt: why did that bug happen, and how would you spot it faster next time?	• Challenge them to test their game against every line of their design brief and list what differs. • Ask them to modify and re-test one feature — a faster projectile or a different scoring value — then explain the effect to another pair.

Cross-curricular hook

Links to Maths — pupils use the score and health variables as changing numbers, adding by 1 and subtracting by 10 as the game runs.