LESSON 15

Monster Battle Arena

60 minutes | Game Design Studio

WHAT THIS LESSON DOES

Get ready to build an exciting monster battle game using MakeCode Arcade! You'll create a player sprite and an AI-controlled monster, design combat mechanics, and manage health systems. By the end, you'll have a game to play and share with friends.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Develop a player-controlled sprite and an AI-controlled monster in a game using MakeCode Arcade.
- Create and manage a new project in MakeCode Arcade.
- Implement a health system for player and monster sprites.
- Develop a combat system where player and monster sprites can inflict damage on each other.
- Implement a win/lose condition based on the health of player and monster sprites.

BEFORE THE LESSON

- Open and test https://arcade.makecode.com/ on the school network beforehand; confirm the site loads and is not blocked by the filter.
- One device per pupil, charged, with sound on so hit effects and camera shake register.
- Decide whether pupils sign in to save projects or work in a single session; have any login details ready if they do.

Monster Battle Arena

HOW THE LESSON RUNS

Introduction
Create New Project
Create your player
Create AI Monster
Making the Monster Move
Health System
Implementing the Combat System
Shooting the Monster
Monster Damage
Determining the Winner
Game On!

TEACHING MOVES

- 2. Create New Project. Have everyone name the project 'Monster Battle Arena' the same way so you can spot who is on track at a glance as you circulate.
- 3. Create your player. Point out this sprite is built from an image grid, not typed line by line. Show pupils where the sprite editor opens so they draw rather than copy pixel codes.
- 4. Create AI Monster. Stress that the monster is a second, separate sprite of kind Enemy. Pupils who reuse the Player kind here will break the combat later.
- 5. Making the Monster Move. Explain the 1000 milliseconds means once a second, and that randint gives the random direction. Let pupils change the number and watch the monster speed up or slow down.
- 6. Health System. Have pupils create both variables and set each to 10 inside on start. Check the variable names match exactly, as a typo here silently stops damage working.
- 7. Implementing the Combat System. Walk through onOverlap Player and Enemy as the trigger. Ask pupils to predict what happens when the two sprites touch before they run it.
- 8. Shooting the Monster. Confirm everyone knows the A button is also the space key on the keyboard so they can test fire straight away.
- 9. Monster Damage. Highlight that this overlap is Projectile and Enemy, a different pair to step 7. Have pupils fire and watch the monster's health drop by one per hit.
- 10. Determining the Winner. Explain gameOver(true) means the player wins and gameOver(false) means they lose. Have pupils test both outcomes by editing a health value to trigger each end.
- 11. Game On! Pair pupils to play each other's games and suggest one improvement, such as a second monster or a starting health other than 10.

WHAT TO WATCH FOR

- Pupils give the monster the Player kind instead of Enemy, so the overlap and projectile code never fire. Show them where sprite kind is set and have them check the monster reads Enemy before adding combat code.
- Pupils think the projectile 'knows' to hit the monster, then are confused when nothing happens without the onOverlap block. Explain that shooting and detecting a hit are two separate pieces of code, and the game only reacts because step 9 tells it to.
- A mismatch between a variable name in on start and in the combat code, so health never changes. Have pupils read the variable names aloud side by side; the game does not guess what they meant, spelling must match exactly.

DIFFERENTIATION

- Emerging: Sit them beside a partner who has the project started and let them build the same sprites step by step alongside. Pre-check they are on the arcade site with a new project open before the first coding step so they are not stuck at the starting line.
- Developing: Ask them to explain back what the onOverlap block does before moving on, so they follow the logic rather than only copying blocks. Prompt them to test after each step rather than at the end, so a broken block is caught early.
- Proficient: Challenge them to add a second monster or change the starting health and balance the fight. Ask them to add a power-up or a new effect and explain to you which event triggers it.