

LESSON 24

Maze Madness

60 minutes (2 periods)

Introduction to Game Design

WHAT THIS LESSON DOES

In this lesson, students will learn to create an engaging arcade game using MakeCode Arcade. They will design a character, control it with buttons, draw a maze for navigation, and set a game goal. The lesson also covers skills like sprite creation, coding for character movement, maze design, camera control, and game-winning conditions.

WHAT STUDENTS SHOULD BE ABLE TO DO

- Develop skills in creating and controlling a character sprite in MakeCode Arcade.
- Understand and apply the concept of drawing and setting up a maze for character navigation.
- Learn to set up and implement game goals using different tiles.
- Gain proficiency in using code to detect sprite overlaps and trigger game outcomes.
- Enhance problem-solving and creative thinking skills through game development.

BEFORE THE LESSON

- Open <https://arcade.makecode.com> on the teacher device and confirm the school filter allows it.
- One charged device per pupil or pair, with browsers ready and a way to save work.
- Projector or large screen helps when showing the tilemap editor and Draw Walls tool.

Maze Madness

HOW THE LESSON RUNS

Introduction
Create a new Arcade project
Create a sprite
Move with the buttons
Draw your maze
Set the walls of your map
Camera follow the sprite
Add a goal tile
Win the game
Wrap up

TEACHING MOVES

- 1. Introduction. Tell the class they will build a complete maze game today: character, maze, walls, goal and a win screen.
- 2. Create a new Arcade project. Get every device to arcade.makecode.com and start a fresh project so everyone begins from the same blank canvas.
- 3. Create a sprite. Add the sprite-create code together, then give time in the sprite editor for each pupil to design their own character.
- 4. Move with the buttons. Add the controller move code and have pupils test movement with the buttons before building the maze.
- 5. Draw your maze. Open the tilemap editor and sketch a simple path first. Keep early mazes short so testing stays quick.
- 6. Set the walls of your map. Show the Draw Walls tool (3d wall icon). Pupils click every wall tile, then try to walk through to prove walls work.
- 7. Camera follow the sprite. Add camera-follow code and have pupils walk to the screen edge so they see the view scroll with the sprite.
- 8. Add a goal tile. Pick one distinct tile as the goal and place only a single copy somewhere reachable in the maze.
- 9. Win the game. Add `onOverlapTile` for `SpriteKind.Player`, match the exact goal tile they used, then play until You Win appears.
- 10. Wrap up. Invite a few pupils to demo their maze. Ask one thing they would change or add if they had more time.

WHAT TO WATCH FOR

- The maze looks solid but the sprite still walks through wall tiles. Reopen the tilemap, select Draw Walls, click every wall tile again, then retest movement.
- The sprite reaches the goal tile but nothing happens and the game does not end. Check that onOverlapTile uses the same tile they placed as the goal, not a different dungeon tile.
- The character vanishes when it moves past the edge of the screen. Confirm the camera-follow block is in the project and attached to their player sprite.

DIFFERENTIATION

- Emerging: Pair with a partner for the first code blocks and sprite editor steps. Suggest a very short maze with few wall tiles so they can reach a working win quickly.
- Developing: Prompt them to test after each change: movement, then walls, then goal overlap. Have them explain which tile is the goal and how the overlap block matches it.
- Proficient: Challenge them to design a longer maze path using only the tiles already in the editor. Ask them to place the goal further away and retest walls and camera follow.