

Lunar Lander Scratch Game

In this activity, you will learn the fundamentals of coding in Scratch by creating a simple game. In the game, players pilot a lunar lander on their way to their moon. Players must control direction and thrust to land on target and beat the game. As you create the code for the Lunar lander sprite, you will become familiar with the Scratch code blocks, learn how sprites interact with each other, and finish with a playable game.

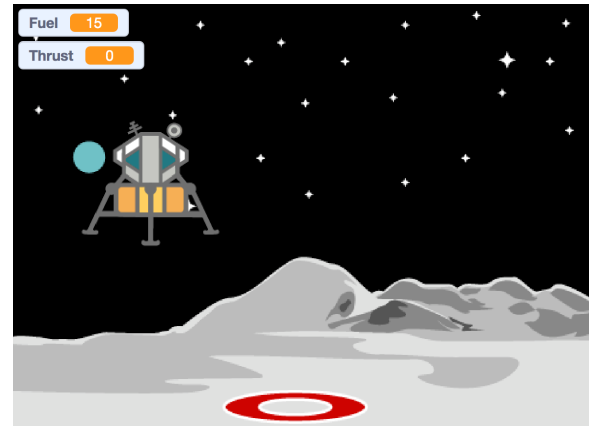


Figure 1
Landing on the moon!

Getting Started

1. Open the Scratch editor window at <http://scratch.mit.edu/projects/editor>
2. From the File menu, choose "Load from your computer".
3. Choose the "Lunar Lander Scratch Game Student.sb3" file and open it. (Lunar Lander Scratch Game Instructor.sb3 is a completed project for reference.)

The Scratch window is divided into three main sections.

- A. Stage and Sprites:** The Stage is where the action happens! Below the Stage, you'll see all the sprites used in your Scratch project. Sprites can be anything from game elements to story characters to user instructions.
- B. Sprite Code:** The center section is where you create programs for each sprite.
- C. Block Palettes and Code:** The interlocking blocks on the left, organized into palettes like "Motion", "Looks", and "Variable", is the code you use to control sprites and the Stage. You drag blocks from this section into the Sprite Code section to create your program.



Figure 2
The Scratch window

By starting with the Lunar Lander Scratch Game.sb3 file, you will already have two sprites and a stage.



A moon backdrop for the Stage



An incomplete Lunar Lander sprite: This sprite is controlled by the player. You will need to create code to place it at a starting position and to move it from side to side as it drops towards the moon.



An incomplete Target sprite: This will be the landing pad. You will create code to place it on the surface of the moon at random locations.

Lunar Lander Sprite Code



Let's start by creating the code for the Lunar Lander. Click on the Lunar Lander sprite in the Sprites area below the Stage.

The code works like this:

1. Use a hat block, like the "When green flag clicked" hat block shown here, to program when Scratch runs. In this case, you're using the green flag to start the game. The code that is connected to this hat block will run after the player clicks the flag. First, there are three motion blocks that place the lunar lander at $y = 131$, $x =$ some random position, and pointed straight up. The "Thrust" and "Fuel" variables are set with their initial values of 0 and 15, respectively. A "repeat until" control block is then used as a loop. In this loop, the lunar lander moves up by a value of Thrust and down -10 . If Thrust is 0, the lander drops by -10 each loop. If Thrust is 10, the lander will not rise or drop. This loop will continue until the y location of the lander is less than -165 .
2. The Thrust and Fuel variables and blocks, which have already been created for you, can be found in the Variables palette. The "Thrust" and "Fuel" variables are set with their initial values of 0 and 15, respectively.
3. The four other hat blocks, and the code connected to them, are used to control the up, down, left, and right movement of the Lunar lander. First the code checks to see if there is any fuel left. If there is, then either the Thrust (which changes the y position of the lander) or the lander's rotation are modified. Each time the modification happens, the Fuel variable is decreased by 1. Since the program starts with $Fuel = 15$, the player has 15 attempts to modify the Thrust or rotation.

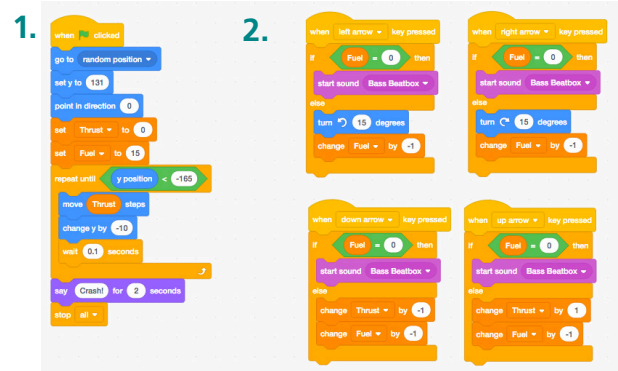


Figure 3
Lander sprite programs
(See page 4 for larger screenshot)

After creating this code, try it out!

- Click the green flag. Does the Lunar Lander start at the top of the stage?
- Can you move it around the stage by using the left, right, and up arrow keys?

Target Code



Next, let's create the code for the Target sprite.

Click on the Target sprite in the Sprites area below the Stage.

The code works like this:

1. The target is a simple block that is triggered by the player when the green flag is clicked to get started. This block randomly selects an x position where $y = -155$. This places the target near the bottom of the stage in a random position on the x axis.
2. A "go to back layer" block moves the target to the back layer, a layer behind the Lunar lander. This ensures that as the Lunar lander passes in front of the target it appears to land on the target, rather than disappear behind it.
3. The "say You did it! Hooray! for 2 seconds" block creates a speech bubble, if the Lunar lander finishes on the target. You will have successfully completed the mission.



Now that you've created all the code, try it out!

Click the green flag to start your game from the beginning. Can you land your Lunar lander on the target?

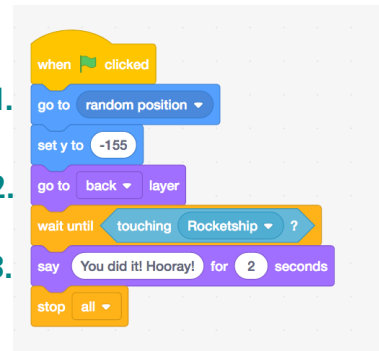


Figure 4

Target sprite code

(See page 4 for larger screenshot)

Challenge Extensions

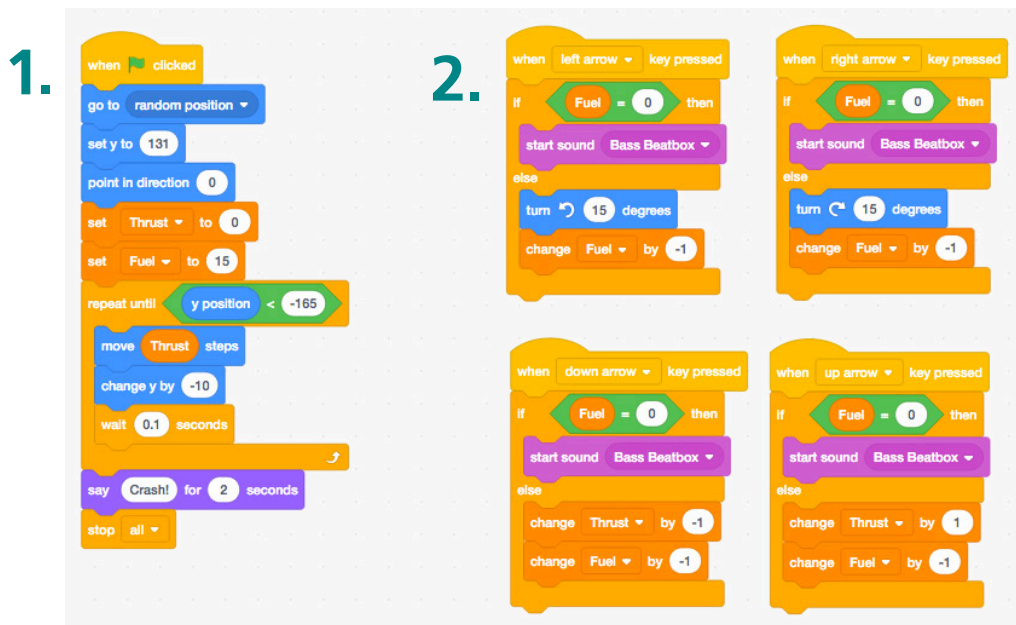
Try adjusting the values in the "Thrust" block to make the Lunar lander sprite move faster or slower. Originally thrust is set to -10 . Change to -5 or -12 values and see how it affects the speed of the Lunar Lander sprite as the game begins.

1. Try controlling the Lunar lander with a Go Direct® Force and Acceleration sensor. Click on the Extensions icon in the lower left corner. Select the Go Direct Force and Acceleration extension. Be sure to have the Scratch Link software running and your Force and Acceleration sensor turned on; a window will pop up allowing you to choose your sensor from a list. Once you have added the extension and connected to their sensor, they'll have a whole new set of blocks. Swap out the "when left arrow key pressed", "when right arrow key pressed", "when up arrow key pressed", and "when down arrow key pressed" hat blocks for the "when tilted left", "when tilted right", "when tilted up", and "when tilted down" hat blocks. Now, when you tilt the sensor left, right, up, or down the Lunar lander changes direction or thrust!

Students might have to modify how fast the Lunar lander is falling. In addition, keep in mind that these hat blocks respond to a tilt from a neutral position. This means students have to tilt, bring back to neutral, and then tilt again. They cannot just hold the sensor in a tilted position.

2. Can you think of any other variations? What other variables can be introduced to make the game more difficult?

Figure 3
Lander sprite programs



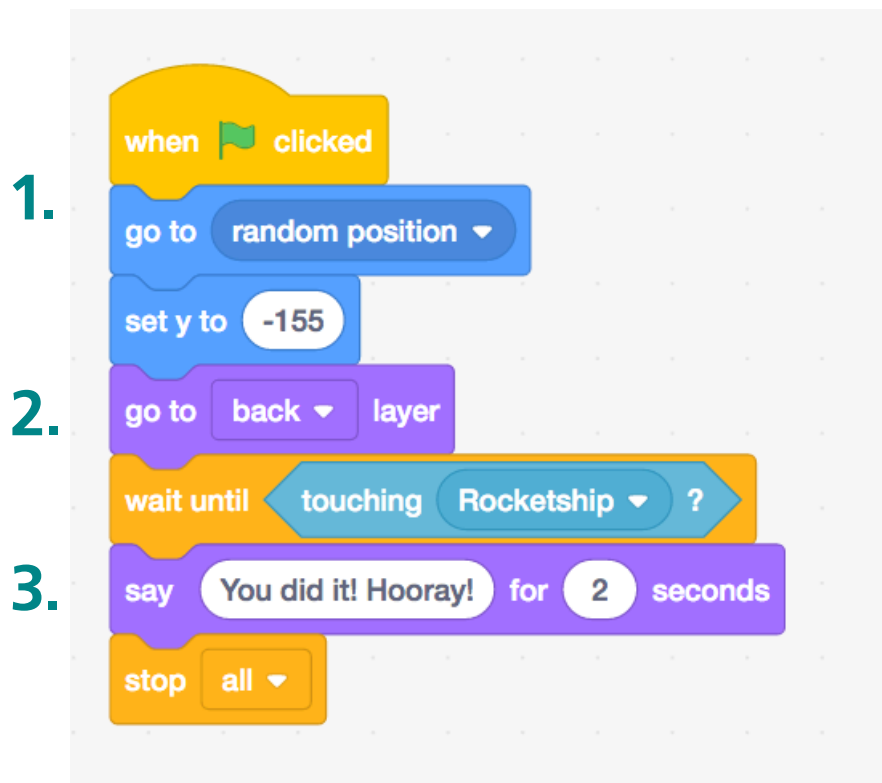
1. **Initial Landing Sequence:**

- when green flag clicked
- go to random position
- set y to 131
- point in direction 0
- set Thrust to 0
- set Fuel to 15
- repeat until y position < -165
 - move Thrust steps
 - change y by -10
 - wait 0.1 seconds
- say Crash! for 2 seconds
- stop all

2. **Control Sequences:**

- Left Arrow:** when left arrow key pressed, if Fuel = 0 then start sound Bass Beatbox, else turn 15 degrees and change Fuel by -1.
- Right Arrow:** when right arrow key pressed, if Fuel = 0 then start sound Bass Beatbox, else turn -15 degrees and change Fuel by -1.
- Down Arrow:** when down arrow key pressed, if Fuel = 0 then start sound Bass Beatbox, else change Thrust by -1 and change Fuel by -1.
- Up Arrow:** when up arrow key pressed, if Fuel = 0 then start sound Bass Beatbox, else change Thrust by 1 and change Fuel by -1.

Figure 4
Target sprite program



1. **Initial Positioning:**

- when green flag clicked
- go to random position
- set y to -155

2. **Targeting:**

- go to back layer
- wait until touching Rocketship

3. **Completion:**

- say You did it! Hooray! for 2 seconds
- stop all