PROGRAMMING IN ALICE

ALICE WORKSHOP: EVENTS AND EVENT HANDLING

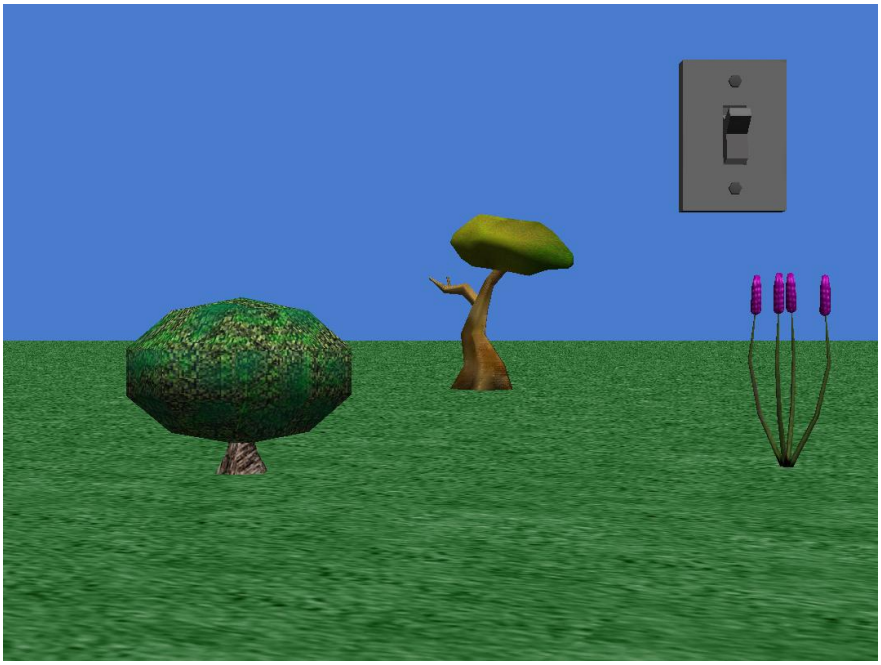
LEARNING OBJECTIVES

The learning objectives for this workshop are:

- Students can design and implement event handling methods
- Students can use events to implement interactive programs

EXERCISE 1

Create a new world that includes a light switch (from the Controls gallery), together with some other scenery, as shown in the picture below.



In this exercise you will use an event to "turn off the light" when the switch is pressed.

1. Create a new property variable, turnedOn,with a Boolean type, for the lightSwitch object. Initially this variable should be true.
2. Create a new method, named toggleSwitch, for the lightSwitch object.
    a. The method should use an if statement to test whether or not turnedOn is true.
    b. If it is true, then turnedOn should be set to false, the light switch subpart should be turned forward so that it is in an on position, the light object brightness

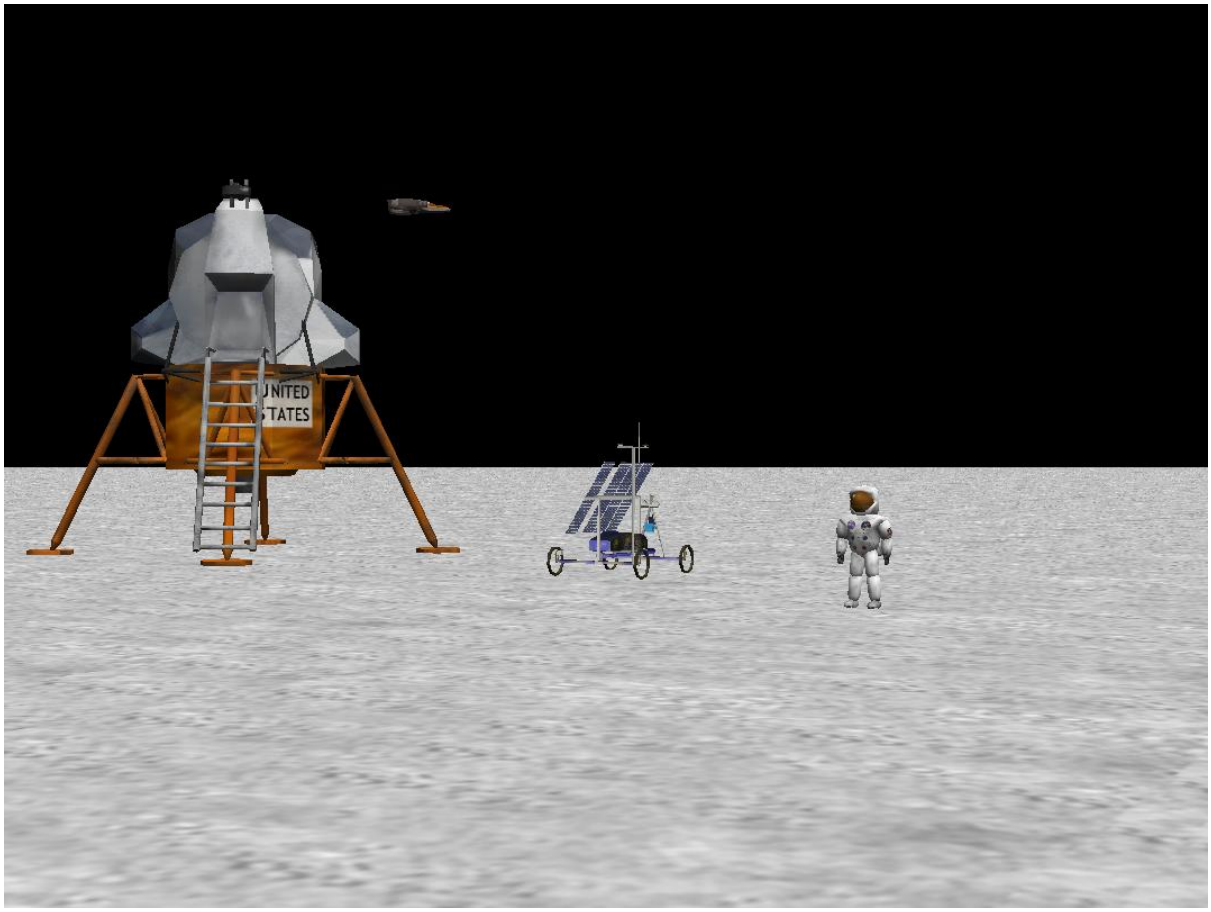
property should be set to 0, and the world object atmosphere colour property should be set to black.

c. If turnedOn is false, then set turnedOn to true, return the switch subpart to its original position, and reset the light brightness and atmosphere colour to their original values.

3. Add a new mouse is click on object event. The subject of this event is the lightSwitch object (or you may select the switch subpart). This event should call the toggleSwitch method. Run your world to ensure that it works correctly.

EXERCISE 2

Create a new world using the space template. The world should include astronaut, lunarLander and hyperionRobot objects (from the Space gallery). You should also include a grayJumpJet object (from the SciFi gallery), placing it in the background. Your world should look like the one shown below:



In this exercise you will use a variety of different events to interact with the objects in the world.

1. Create a new method, **spacewalk**, for the astronaut object. This method simulates a spacewalk, moving the astronaut up then down, at the same time as moving the astronaut forward. You will need to use a combination of do together and do in order commands.

2. Create a new **when key is pressed event** so that when the Enter key is pressed, the spacewalk method is called. Run the world and test whether the event handler works by pressing Enter a number of times.
3. Create a new method, **starjump**, for the astronaut object. This method will make the astronaut do a star jump. The astronaut will move upwards and at the same time moving his/her arms outwards to form a star shape. The astronaut should then return to the ground, with his/her arms and legs returning to their starting position.
4. Create a new when key is pressed event so that when the Space key is pressed the astronaut performs a star jump.
5. Create a new **let arrow keys move subject** event to move the hyperionRobot object.
6. Create a new method, **takeoff**, for the lunarLander object. This method should move the lunar lander up 1 metre. Change the style for the move command to *abrupt*.
7. Create a new **while key is pressed** event. This can be done by creating a **when key is pressed event** and then right clicking on the event and changing to **while key is pressed.** Select the W key, and select the **takeoff** method for the event action. Run the world to ensure that it works correctly.
8. Repeat the previous two steps, this time creating a **landing** method which is called while the S key is pressed. Note: you will need to include an if condition to ensure that the lunar lander does not move below the ground.

EXTENSION EXERCISES

9. Use a while world is running event to move the **grayJumpJet** object back and forth across the background continually. Hint: this can be done by calling a method that moves the jet across the screen and then turns the jet around.
10. Add simple collision detection for the hyperionRobot object.
    a. Develop a Boolean valued function, **collision**, for the hyperionRobot, which returns true if the hyperionRobot is within 5 metres of the lunarLander.
    b. Create a **while something is true** event. Use the collision function as the condition. Move the hyperionRobot backwards ½ metre, abruptly in zero seconds. Note: this will only work correctly if you drive the robot forward toward the lunar lander.
    c. Update the collision function by using the width and depth functions to get a more accurate measure of the distance between the lunarLander and hyperionRobot.