SPRK
schools parents robots kids

OrbBasic Lesson 3
Sensors and Random: Teacher Guide

Overview

Students will use Sphero to explore the computer science concepts of reading sensors and storing data in variables. They will use OrbBasic, which is a text-based programming language for the Sphero. They will write a simple program that detects when Sphero is in the air and also when Sphero collides with an object and then when these events happen, Sphero will change color or behavior.

In OrbBasic Lesson 2 students learned about conditional statements such as if/then/else. In this lesson, students will use these conditional statements in conjunction with the function that reads the accelerometer data in order to build a program that reacts to sensor readings. Students will also use knowledge from Lesson 2 about changing Sphero's color using LEDC. At the conclusion of this lesson students will work with the random function to make Sphero move at a random heading after a collision.

Read through the student guide to learn more about the accelone and random functions. At the start of the lesson, discuss with students about acceleration and accelerometers.

Objective

Students will:

- Create a short OrbBasic program that changes the color of Sphero when it senses that it is in the air.
- Create another OrbBasic program that recognizes when Sphero collides with something and then changes color.
- Create a third OrbBasic program that sends Sphero in a random direction after a collision.

Common Core Math Standards

The following Common Core Math Standards for 4th and 5th grade apply to this lesson:

- CCSS.MATH.CONTENT.4.OA.C.5: Generate and analyze patterns
- CCSS.MATH.CONTENT.5.OA.B.3: Analyze patterns and relationships
- CCSS.MATH.PRACTICE.MP1: Make sense of problems and persevere in solving them.
- CCSS.MATH.PRACTICE.MP2: Reason abstractly and quantitatively.
- CCSS.MATH.PRACTICE.MP4: Model with mathematics.
- CCSS.MATH.PRACTICE.MP8: Look for and express regularity in repeated reasoning.

Materials Needed

Spheros are controlled via Bluetooth on either Apple (iPod, IPhone, or iPad) or Android devices. Ideally, you would do this lesson in groups of 3 or 4 students, each with their own Sphero and device. This lesson is designed for iPads, but other devices could be used. Here is what each group would need:

Materials Needed (continued)

- iPad with Sphero OrbBasic loaded. You can get Sphero OrbBasic for free from the iTunes app store.
- Sphero that has been fully charged
- Print-out of the worksheet
- A flat open space. (Preferably not very slippery.)
- Objects to have Sphero collide with

  Although not required, it can be helpful to have a keyboard attached to the iPad.

Part 1: Connect the Sphero

In part 1, students need to connect each iPad with a Sphero. They will:

- Wake up the Sphero
- Turn on Bluetooth
- Connect the correct Sphero to the iPad, using the colors that it flashes as a way to tell which Sphero has which name

Part 2: Aim the Sphero

In part 2, students need to set the orientation, which is the direction of 0 degrees heading for Sphero. This is called "aiming". It's important that they get this right so that the Sphero will follow the path and not bump into anything. To do this, they need to adjust the blue"taillight" so that it is pointing directly at them. If they do this correctly, then the Sphero will roll directly away from them. Students will:

1. Open up OrbBasic on the iPad
2. Hold the Sphero in front of them as they look down the path
3. Tap and hold the aim icon at the bottom of the screen and adjust the taillight so that it is pointing directly at them.

Part 3: Sensing when Sphero is in the Air

In part 3, students will create an OrbBasic program that changes color when Sphero's accelerometer senses that Sphero is in the air. The new command students will use in this part is called accelone. Accelone is set to be 0 for a perfectly calibrated Sphero in freefall (if the only acceleration acting on it is gravity). Due to the fact that few things are perfect, students will need to build in a soft buffer to determine if Sphero is in fact in the air. This will be done by coding if accelone< 200 instead of if accelone=0.

Students will use a conditional statement and the LEDC commands to change the color of Sphero when it is in the air. If the condition is false then the program will loop back to the beginning. See the student guide for the code.

Part 4: Sensing Collisions

For Part 4 students will create a macro similar to the one they built in part 3 but it will be sensing collisions instead of whether or not Sphero is in the air. Students will be building a code to read if the accelerometer value is greater than 5000 (which indicates that Sphero has collided with something).

Part 4: Sensing Collisions (continued)

If this conditional statement is true the program will move down to the next line of the program and follow a set of commands that change Sphero's color to red for 1 second. However, if the conditional statement is false then the program will utilize a goto statement that goes back to the beginning of the program. After building the program students can use the joystick in OrbBasic to test the program, it should turn red when it collides with another object.

Part 5: Challenge

In part 5, students will see if they can build a code that makes Sphero drive straight and then move in a random direction after a collision. The heading range for Sphero is 0-359 degrees, instead of 1-360 degrees. To ensure the program doesn't receive an error, we have to make the highest possible value for random to generate 359. In this program students may want to use line 5 to set the initial heading to 0 for Sphero. Using lines be useful because we only want the heading to be set once at 0 as an initial condition and then the program could then loop through the commands starting at line 10 as in the other macros.

For the challenge, ensure you have some open space in your classroom with a few obstacles set up. When planning which objects Sphero will run into, try to avoid using things like walls because it will be harder to see what direction Sphero travels after the collision. The answer to the challenge is below (the line numbers and LEDC values don't have to be exactly the same):

```
5 h=0
10 LEDC 2
20 goroll h, 100, 1
30 if accelone> 5000 then goto 40 else goto 10
40 LEDC 1
50 heading rnd 359
60 delay 500
70 goto 10
```