ELEMENTARY COMPUTER SCIENCE WITH GAME DESIGN

TEAM COLOR SEARCH

LESSON 4

Summary

Ozobot bounces off of two teammates' hand-held colored paper in a race against time until it reaches red. Students learn variables for tracking time and have Ozobot report the total.			
CS Topics: Loops and breaks, conditional logic, sensor input/output, + math, + variables		**Learning Outcomes:** Build a time counter, discover how to build an interactive game and 3D map.	
Grades: 3-5	**Coding Level:** Intermediate	**Map Style:** Free Movement	**Game Mechanic:** winning, +timing, +multiplayer

Introduction

Welcome to the fourth lesson in Ozobot's Elementary Computer Science with Game Design! Students will quickly develop their coding knowledge and intuition by 'deconstructing' a game's program to see what type of code creates each action, then create their own program. Leaders in computer science teach themselves new skills through exploring and modifying sample programs this way. These lessons are also great for teachers still learning coding, themselves. Read the Elementary CS with Game Design README for more about this series.

This game is all about teamwork. We use the concept of color reactions to make a game where students use colored cards slipped under a moving Ozobot to change its direction around 3D and color obstacles. The goal is to get Ozobot to the winning color. Once Ozobot reaches red, it will report how many (approximate) seconds it took.

To play, one player will hold the blue card, which makes Bit or Evo turn a slight left, and the other the green, to turn slight right. (If you're standing at the start circle side of the map and looking towards the winning red, the player on your left has the green card and the player on your right has the blue card. When students explore the program, they can add speed changes as reactions to the colors, too.

This game is set against time. We introduce the concept of variables, a way to save and update a number value in code, to create a timer. We also make Bit and Evo tell players their play time at the end.

After playing and exploring this game, students can explore building their own maps and creating new abilities for their robots. Using the DIY 3D interactive map idea, students can explore more game ideas, or coordinate a dance for their bots, or anything they can dream up!

To get started, read each section of this lesson, below, before teaching the lesson. SETUP lists the preparation steps needed for this lesson, ABOUT THE CODE explains the programming concepts used, and LESSON OUTLINE is a simple step-by-step walkthrough for teaching this lesson. We recommend the teacher tries out the program and map before class to clarify the coding concepts and game play.

Questions or comments about this lesson? We'd love to hear from you! Email us at ozoedu@ozobot.com

SETUP

'TEAM COLOR SEARCH' OZOBLOCKLY PROGRAM ozo.bot/team-color-search

REQUIRED MATERIALS

- **FOR DEMO:** 1 Bit or Evo (pre-programmed, link above) & 1 prebuilt map,
- 1 Bit or Evo per group,
- Map materials (paper, tape, Ozobot color markers, 3D objects),
- 1 computer or tablet with wifi per group,
- 1 printed student worksheet per student (or write Q&A on the board) (attached).

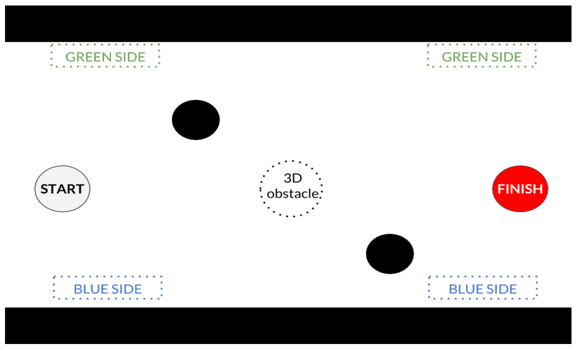
Time-saving tip
Set all devices to the program link, above, and calibrate all bots to the screen before class

STUDENT GROUPING

- Groups of 2-3 students are recommended since this is a collaborative game.

MAP SETUP

- Your table is your map, as long as it's white or a color Ozobot isn't looking for! If not, place down paper and tape all edges of paper together.
- Color two 2"x4" strips of paper completely green and blue. These are the handheld cards.
- Color one red circle (1"-2") for win and several black circles for lose, and tape completely down. You can make an uncolored circle to mark the start, or a white X.
- Place a 3D obstacles right in the middle of the map. Here is a possible setup, on the right.
- Remember to calibrate Ozobot to a black circle before playing.



GAME PLAY STEPS

- **OBJECTIVE**: Get Evo or Bit around 2D color and 3D obstacles to land on the winning color in the least time.
- Start Ozobot on one end of the table. Two players will lay their colored cards in front of Ozobot to navigate around obstacles and onto the red winning space.
- Green is on the Ozobot's left, and blue on its right.

DEMONSTRATION SETUP

- Set up your projector, or a main table, to show your demo so all students can see.

TIME MANAGEMENT

- This lesson is planned for 50 minutes, or two lessons of 30 minutes.

ABOUT THE CODE

'TEAM COLOR SEARCH' OZOBLOCKLY PROGRAM ozo.bot/team-color-search

We are familiar with how Bit and Evo see colors and react to them based on how they're programmed. In this game, we get rid of our map to play with 3D and 2D obstacles on any flat surface so it's more interactive and dynamic. We also keep track of time with a variable, and report that time using lights (sounds on Evo).

LOOPS

Loops allow a section of code to be repeated a certain number of times, or forever until a 'break' code is called. In this program, we use "**break**". This code helps create autonomous movement and the ability to win.

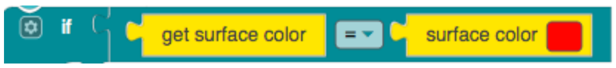
To play these games, we need Ozobot to check the colors underneath it regularly. The best way to do that is to write a short movement code followed by code to check the color underneath (Conditional Logic). By using a loop with these two codes, we can create autonomous movement where Ozobot reacts to a map made of color often.

For winning and losing, we need to have the '**break'** code for both conditions. But, we don't want the same victory dance reaction, so the win/lose reactions have to be inside each condition, instead of one victory dance below the loop. There are two 'break' codes, but only one will ever be called per game. You can, in theory, use as many breaks as you need, since only one will ever run.

CONDITIONAL LOGIC

In programming, sometimes we want some code to run in a specific situation or condition. Programming languages make this easy by giving us "**if**" statements.

In OzoBlockly, we can check for the color of the paper underneath Bit and Evo by using this line of blocks:



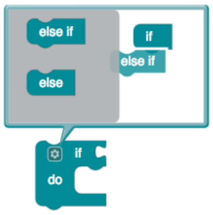
This is basically a sentence: "**if** the color that the color sensor sees **is** red…" This if statement resolves to either true or false. If it's true, Ozobot sees red, then run the code that's with it. If not, skip and do the next code.

The equals sign is the Boolean Logic, named after George Boole, a 19th century mathematician. In this type of logic, the result is either true or false. This is great for programming because the two choices can easily be turned into binary code (1s and 0s, which all what code boils down to).

'Else if' in this program sounds funny to say, but it is just a quick way to say, 'if not the last condition, then if this different condition is true…' You can have as many 'else if's as you want. Students are not expected to edit the 'if' block. If you'd like to know how, use the info box below.

HOW TO EDIT THE 'IF' BLOCK

To add, or remove, 'else if' code in the if block, click on the gear on the block. This opens a sort of mini OzoBlockly.

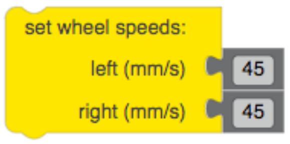


- Drag 'else if' from the left and attach it on the right to add.
- Drag 'else if' from the right and into the left to delete it.
- To close this window, click the gear again.

Students are not required to edit this block in this lesson.

MOVEMENT

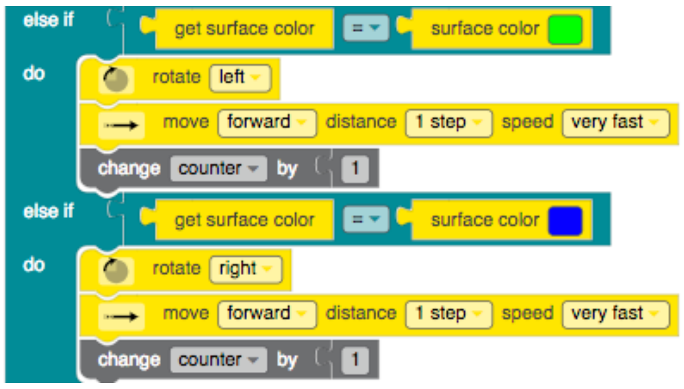
The movement in this program is 'set wheel speeds' from Mode 4 movement. This code is being used instead of a step because it will see any changes in surface color very quickly. If students change the speed of their program, both blocks for left and right motor speeds should be edited. Otherwise, you'll get some kind of turn.


```
set wheel speeds:
    left (mm/s)  45
    right (mm/s) 45
```

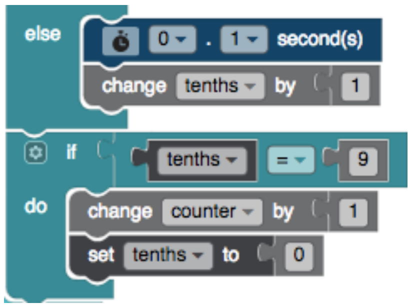
LOOPS

Variables are a way to save data, and update it. In today's program, we a counter to keep track of time. At the end of the program, Ozobot will tell us (with lights or sounds) how many seconds the game took. ("change counter by" block is in Mode 4 Math.)

We need to create our own timer inside of this program, but since OzoBlockly is sequential, we can't use a Timing block for 1 second without making the movement take too long and affect color detection. So, we need to be a little clever.


```
else if  get surface color = surface color 🟩
do   rotate left
     move forward distance 1 step speed very fast
     change counter by 1
else if  get surface color = surface color 🟦
do   rotate right
     move forward distance 1 step speed very fast
     change counter by 1
```

What we will do instead is figure out how much time each movement will take (already done in the given program) and create a counter that way. Each count of the counter is 1 second, which is why the reaction to green and blue is 1 count, or one second, long (above).

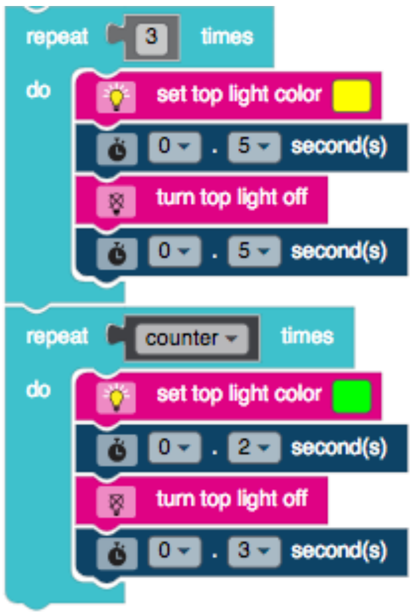


```
else
    0 . 1 second(s)
    change tenths by 1
if tenths = 9
do
    change counter by 1
    set tenths to 0
```

For movement on white, we count every tenth of a second in a separate timer, called ‘tenths’, so we can make sure Ozobot sees any other surface color quickly. Every time Ozobot reaches 10 tenths of a second (from 0 to 9), it adds 1 count to the ‘counter’ variable. This way, we can keep track of all whole seconds of the game.

COUNTER SUMMARY

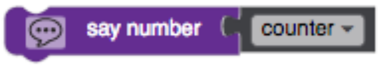
To sum up, each reaction to blue or green is worth a second (1 of the ‘counter’), and movement on the white surface is counted in 10ths of a second, recording every full second as a count on the ‘counter’. This way, our timer is pretty accurate.



```
repeat 3 times
do
    set top light color 🟨
    0 . 5 second(s)
    turn top light off
    0 . 5 second(s)
repeat counter times
do
    set top light color 🟩
    0 . 2 second(s)
    turn top light off
    0 . 3 second(s)
```

REPORTING THE TIME

To tell the time we can have the top LED blink. The code on the right tells the time after a three second pause (three yellow blinks). Count the rapid, green LED flashes, which each represent one second. For Evo, you can use the sound block from Mode 4 ‘say number’ and attach the variable ‘counter’.



```
say number counter
```

You can also devise a way (or have students try) to make one color blink to show the tens place and a second color for the ones place. This could be helpful if games are long.

NOTE: The max time that the timer can report is 127. That’s because the memory in Ozobot for any single variable has that max.

LESSON OUTLINE

DEMONSTRATE THE GAME - 10 minutes

Have one map and one programmed Ozobot ready to demonstrate to the class (see SETUP, above).

1. **Organize** students into pairs or groups.
2. **Explain** today's lesson to your students: *We will play an interactive game with Ozobot that also has a timer. We'll learn how to make our own timer.*
3. **Hand out** one worksheet (attached) to each student (or do the Q&A on a board or screen.)
4. **Demonstrate** the game program as many times as you like, with all possible situations like winning on red, losing on black. A student can help you play one side of the map. Explain how to play: *Start the program and set Ozobot on the starting space, angled away from an obstacle. Ozobot will walk forward, and once it needs to turn, place a colored card in front of it (not shove it under the bot). Ozobot will walk onto the color, see it, then turn and walk off. Keep Ozobot off and away from the obstacles, and get it onto red.*
5. **Students complete** sections A and B of the worksheet. We'll compare the answers to the program next.

DISCUSS HOW THE PROGRAM WORKS - 15 minutes

1. **Open** the OzoBlockly program (ozo.bot/team-color-search) on your main screen.
2. **Point out,** using ABOUT THE CODE, the codes that make Ozobot see color: conditional logic;
   a. Explain that each 'if' and 'else if' is a sentence like 'if the color Ozobot sees is (equal to) the color red, then do…'.
   b. Students do not have to edit any part of the if code; they may just want to edit the 'do' parts.
3. **Point out,** using ABOUT THE CODE, the codes that make Ozobot win or lose: loops;
   a. Point out the two **break** codes for the colors red and black. These both end the loop, and the game. Once this code runs, the code below the loop, which tells the time of the game, will run.
4. **Point out** the codes that built the timer: variables;
   a. The grey blocks called 'counter' and 'tenths' are special blocks that record any number. These numbers can be changed, to help keep time. Every second, the 'counter' variable will change by 1. Starting from 0, counter will count 1, 2, 3, 4… until the game is over.
   b. 'counter' counts every second, and 'tenths' counts every tenth of a second. When there are ten tenths, Ozobot adds 1 second to 'counter'
5. **Point out** the codes after the loop: the time report;
   a. First, Ozobot will blink yellow three times to let you know it's about the tell you the time.
   b. To read the time, count how many times Ozobot's top LED blinks green. This is how many seconds your game was.
6. **Point out** the map, which can be altered in any way, as long as it's possible to maneuver Ozobot around the obstacles. Students could design maps, then trade maps with other groups for added variety.
7. **Allow** for any questions from students.

8. **Summarize** the program to remind students that they can now code autonomous movement, surface color reactions and the game mechanics of Winning and Losing!

STUDENTS EDIT THE PROGRAM – 25 minutes

1. **Brainstorm** ideas for meaningful edits for this program. Things like speed, rotation of turns, and having the points be reported in a way that's easier to see,
2. **Hand out** one computer/tablet, Ozobot and maps materials to each group of students.
3. **Students go to the program link** (or set each device to that page prior to class). Students should load original program to test on their map before making changes to the program.
4. **Students reprogram** their Ozobot to get a more unique game.
5. Groups can challenge one another to compete in the games they built, keeping track of each group's times on a leaderboard.

TEAM COLOR SEARCH

STUDENT WORKSHEET

A. What does Ozobot do?

On green, Ozobot ____

On blue, Ozobot ____

On red, Ozobot ____

On black, Ozobot ____

B. Pseudocode

This 'pseudocode' is a simple English version of the program for the Team Color Search game. Read through, then fill in the blanks with the words you think belong there.

```
Set LED to yellow.
Wait 2 seconds.
Repeat forever:

    Set wheel speeds to 45mm/s.
    If the color Ozobot sees is red,

         ____________________________________________________________

    If the color Ozobot sees is black,

         ____________________________________________________________

    If the color Ozobot sees is green,

         ____________________________________________________________

    If the color Ozobot sees is blue,

         ____________________________________________________________

Repeat 3 times:
   Set top light _______. Wait _____ seconds. ___________. Wait _____ seconds.
Repeat 'counter' times:

   Set top light _______. Wait _____ seconds. ___________. Wait _____ seconds.
```