



INTRODUCTION TO EVO EVO'S FORCE FIELD

SUMMARY

Discover the infinite programming possibilities with Evo by Ozobot during an immersive coding experience. Students quickly develop confidence and excitement by analyzing a complex program for a game to understand its parts, then create their own cool story or game that uses Evo's infrared force field to help it move.

OVERVIEW

Students start off by playing a game with Evo called "Evo's Color Quest" where they move it by waving a hand near its proximity sensors and move it to green, red and blue to collect points. Students never touch Evo, just use its force field that the loaded program created. Everyone finds out how the program in OzoBlockly is close to simple English, which means that all students need to do is plan out Evo's reactions, then set up the right blocks in order - which they do!

In the creating portion of the lesson, students assemble a smaller version of the game program called "Evo's Green Quest", test it, then modify the program based on planned goals for their own unique game or story. Last, students share their creations.

This lesson was created especially for the Hour of Code™, and can be used for more than the intended hour. Once students have caught the programming bug, get them started on OzoBlockly Basic Training (portal.ozobot.com/lessons/compilation/ozoblockly-basic-training).

SO, HOW DO EVO'S SENSORS WORK?

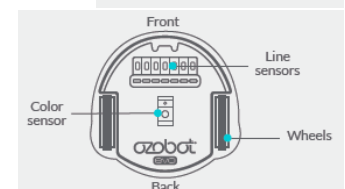
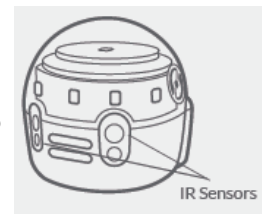
PROXIMITY

Evo has 4 infrared (proximity) sensors around its body – two in back, two in front. They work by sending out an infrared light and looking for it when it bounces back. Evo knows how far something is based on what is bounced back. We recommend obstacles to be light-colored because IR light bounces back best on it.

NOTE in OzoBlockly, setting proximity sensors to 0 looks for nothing, 5 detects objects very far away, and 60 is pretty close. We recommend "≥ 20".

COLOR

Underneath Evo you can see a row of sensors – those are line sensors. They detect the contrast of light and dark. In the middle of the bot is the color sensor. When Evo walks, a light turns on near the color sensor for it to see the color better.



LESSON OUTLINE

1. Students play a pre-programmed game where they help Evo reach the colors on a map using its force field, and analyze Evo's actions on a worksheet.
 - a. Everyone discovers how their analysis looks like the actual program!
2. Students learn to use the editor and assemble a shorter, similar program from scratch, and practice loading programs through Flash Loading.
3. Finally, each student plans their own game or story based on this program.

GRADE LEVEL

Grades 6 and up

DURATION

1 hour

PREREQUISITES

We recommend educators and students have used OzoBlockly or have practiced with ShapeTracer (games.ozoblockly.com) and/or the OzoBlockly Mini Lesson (portal.ozobot.com/lessons/detail/ozoblockly-mini). Experienced coding students and educators will also have fun with this creative lesson!

GROUPING

Individuals or groups (2-3 students). To learn how to program in pairs or groups, watch Code.org's video on Pair Programming (<https://youtu.be/vgkahOzFH2Q>), or use our Pair Programming lesson (portal.ozobot.com/lessons/detail/ozoblockly-basic-training-3).

MATERIALS

- Computer or tablet, 1 per group
- Evo by Ozobot, 1 per group
- Pens or pencils
- "Evo's Color Quest" game map, 1 per group (choose either colored circles or empty circles, to be filled in with markers)
- *(Optional) Ozobot markers*
- Printed Student Handouts 1 and 2, 1 per student
- *(Optional) Printed of Quick Teacher's Lesson Cues*
- *(Optional) OzoBlockly Guide* files.ozobot.com/stem-education/ozoblockly-getting-started.pdf

QUESTIONS ABOUT THIS LESSON?

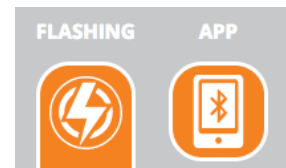
Please contact us at ozoEdu@ozobot.com

LESSON

PREPARATION

Each Evo needs to be loaded with this program (see link) before the class ozoblockly.com/editor?robot=evo#fk3ps8. You could Flash Load them individually, or you can use Bluetooth with your device. To use Bluetooth, follow these steps:

1. Open the link for the complete program, and log in to your Ozobot Evo account (the same as the Ozobot Evo app) in the top right of OzoBlockly.
2. Turn on your phone or tablet with the Ozobot Evo app and log in.
3. Turn on and connect to each of your Evos. The bots might get noisy because the app is tickling them!
4. In the OzoBlockly editor, click the Bluetooth symbol next to the Flash Loading symbol (see right).
5. In the Ozobot Evo app, go to OzoBlockly, and click "Load and Run". Each bot will download the program. Test each bot to make sure it has the program.
6. Fully charge the bots before the lesson.



Set up your TV or projector to show your screen to the class. Alternatively, you can print a copy of the full program (see below) in color to share.

Print one map per group and handouts for each student.

PART 1 - EXPLORE EVO'S COLOR QUEST PROGRAM - 20 MINUTES

Ask students some initial questions on what they think about programming: "What's the hardest part of creating a program? What would you make if you could program robots or computers?"

Explain to students, "Today you will explore a programmed game for Evo and find out how quickly you can learn a complex program. We will start with the game already programmed into Evo. To play the game, place your hands near Evo to move it to each colored circle, starting from black."

Hand out the preloaded Evos and copies of the game map. (If you chose the uncolored version, also hand out markers to color in each circle. We recommend black and green on opposite corners for the later programming activity.)

Walk students through:

1. Turn Evo on and off by clicking the power button once.
2. Calibrate Evo's sensors to see the paper by holding the power button for 2

seconds, let go, and place on the black circle. (Evo turns off after calibration.)
You only have to calibrate sensors to paper once for the day!

3. Play the program by double-clicking the power button quickly while it's on, and place it on the black circle to start.
4. Use your hands near the proximity infrared sensors to get Evo to the three other colors. Neighbors can even race!
5. Students can play the game a few times (double click the button when on) to better understand it. If there are groups, each student should get a turn.

Give all students **Student Handout 1**. Students complete the handout based on what they can see Evo doing while playing the game. Give students 5-7 minutes.

Once students have completed their handout, present the full program to all students on a main screen. You can choose how to analyze it as a class: walk through as a class, ask students for each section's role, or allow students to read it silently. You can give out the screenshot, but take it back so it's not confused with other handouts.

Point out the following:

1. The program blocks are sequential, which means that each one runs after the other and not at the same time.
2. The 'repeat forever' loop means Evo is repeatedly checking its proximity sensors and the color beneath it, but sequential codes mean it can't check its sensors while performing a movement.
3. Each code block's color corresponds to its category's colors on the left, e.g., yellow for Movement, pink for Lights, green for Proximity sensors.

Prompt students to discuss their confidence level in computer programming now, and what they believe they could make and what they need to learn to do that.

PART 2 - ASSEMBLE THE PROGRAM - 20 MINUTES

Tell students, "You will assemble a simpler game program, then modify it to create your own game or story."

Play the OzoBlockly Orientation video: <https://youtu.be/fwlrAzZfvRc> Emphasize the colored categories for codes, and the 5 different modes you can choose from.

Hand out **Student Handout 2**, which contains a program they must assemble from the categories.

When students have assembled their program, walk students through screen calibration and loading:

1. Make sure students chose "Evo", not "Bit" in the top left of the editor.
2. Open the Flashing icon in the bottom left of the screen.
3. To calibrate the sensors: hold Evo's power button for 2 seconds, then release. Evo's wheels will spin. Immediately place the bottom of Evo on the white outline. You only have to calibrate to the screen once today.
4. Turn Evo on.
5. To load: place the bottom of Evo on the white outline and hold. Click "Load Evo" and continue to hold Evo steady on the screen until loading is complete. Evo should be blinking green the entire time. If it's not, try again.
6. For troubleshooting and tips, use files.ozobot.com/stem-education/ozoblockly-getting-started.pdf

Once successful, students can now play their game by getting Evo from the black circle to the green circle. Use the same game map as before.

PART 3 - CREATE A PROGRAM - 20 MINUTES

Armed with newfound skills in OzoBlockly and Evo, students now modify the program to make a new game or a story.

Students follow the rest of **Handout 2** where they plan their new game or story. Our only advice is to use only one light animation (police lights, rainbow) because they make loading the program take much longer.

Depending on how much time is left, give students one or two of these tasks:

1. Completely design their new program to fit the parameters they've outlined
2. Load and test this new program with Evo
3. Write down their reflections about their activity for the day
4. Create a new map or scenery for their creation

If anything isn't completed in class, students can bring their materials (and saved program, see below) home to build their new game as homework, or in another class session. Students can also bring their worksheets home and visit OzoBlockly.com on their own even if they've completed the task.

If students complete their program in class or at home, let them know they can save it by clicking the 'Save' icon, and copying down the new URL link for the page (the link will now have a '#' and a unique six number and letter sequence, like ozoblockly.com/editor?robot=evo#fk3ps8).

INTRODUCTION TO EVO – EVO’S FORCE FIELD QUICK TEACHER’S LESSON CUES

PREPARATION

1. **Load** all Evos with ozoblockly.com/editor?robot=evo#fk3ps8 and fully charge.
2. **Set up** classroom screen to show videos and Evo’s Color Quest program.
3. **Print** 1 map per group, and handouts for each student.

EXPLORE “EVO’S COLOR QUEST” GAME - 20 minutes

1. **Ask** “What’s the hardest part of creating a program? What would you make if you could program robots or computers?”
2. **Hand out** “Evo’s Color Quest” map, and markers if necessary.
3. **Explain** “Today you will explore a programmed game for Evo. To win the game, place your hands near Evo to move it to each colored circle, starting from black.” Demo if necessary.
4. **Hand out** preloaded Evos and walk class through:
 - a. turning Evo on and off.
 - b. Walk through paper calibration (only once for the day).
 - c. Walk through running program (double press while Evo is on).
5. Let students **play**, and maybe race, for 5 minutes.
6. **Hand out** “Student Handout 1”, all students complete in 5-7 minutes.
7. Walk through the “Evo Color Quest” program on the classroom screen.
Explain:
 - a. that their responses on the worksheet looks like the real program,
 - b. that the code block’s color corresponds to OzoBlockly categories.
 - c. that the code blocks go in order and not at the same time (“sequentially”).
8. **Ask** “Has your opinion on the difficulty of coding changed? What could you build?”

ASSEMBLE “EVO’S GREEN QUEST” – 20 minutes

1. **Explain** “You will assemble a shorter game program to get Evo from black to green, then modify it to create your own game or story.”
2. **Play** the OzoBlockly Orientation video <https://youtu.be/fwlrAzZfvRc>
3. **Hand out** “Student Handout 2” and give 5-7 minutes to assemble, or do it as a class if necessary.
4. **Explain** and walk class through:
 - a. Calibrating Evo to the screen (only once for the day).
 - b. Turn Evo on, then Flash Load the program onto Evo.
 - c. Test the game to get Evo from black to green.

Tell students to click the Save icon and copy the new URL link for the page.

CREATE A GAME OR STORY – 20 minutes

1. Students **follow** the rest of their 2nd handout:
 - a. write a goal for their game or story and what the program should tell Evo to do.
 - b. modify the program to create the new version, reload Evo and test.
 - c. write out their reflections.
2. Students **share** the new creation to the class.

Evo's Force Field - Student Handout 1

What's happens when you play with Evo?

If my hand is behind evo, then...	
If my hand is on the left of Evo, then...	
If Evo walks on white, then...	
If Evo walks on red, then...	
If Evo walks on all of the colors, then...	
Describe the victory dance actions in order.	

Here's a section of the program. What is this code telling Evo to do?


```
if surface color is red
do
  set red to 1
else
  if surface color is green
  do
    set green to 1
  else
    if surface color is blue
    do
      set blue to 1
set points to red + green + blue
say number points
```

Write your explanation here...

Evo's Force Field - Student Handout 2

```

    [Lightbulb] set light color [Green] [Green] [Green] [Green] [Green]
    [Speech bubble] say color [green]
    repeat forever
    do
    [Gear] if [object behind]
    do
    [Arrow] move [forward] distance [1 step] speed [medium]
    [Gear] if [read proximity sensor [left front] >= [20]]
    do
    [Rotate] rotate [slight right]
    [Gear] if [read proximity sensor [right front] >= [20]]
    do
    [Rotate] rotate [slight left]
    [If] if surface color is [Green]
    do
    [Break] break out of loop
    stop motors
    [Lightbulb] set front lights color randomly
    [Spin] spin [left]
    [Speech bubble] play [surprised]
  
```

- 1 Go to ozoblockly.com/editor
- 2 Choose "Evo" in the top left: 
- 3 Find the blocks by **color** and in different 'modes': 1 2 3 4 5
(Most are in "3"; others are marked)
- 4 Assemble the program.
- 5 Flash Load Evo and test on your Color Quest map.

(Change to ≥ 20) **Mode 4**

Mode 2

When you've modified your program, **save it!** Click the "Save Icon" to save it, and copy the new URL. ozoblockly.com/editor?robot=evo#_____

Modify the program

What do you want to create? What code do you have to write to make Evo perform that way? *Make a shark hunt story... If an object is in front... If surface color is red... set front lights to all red.*

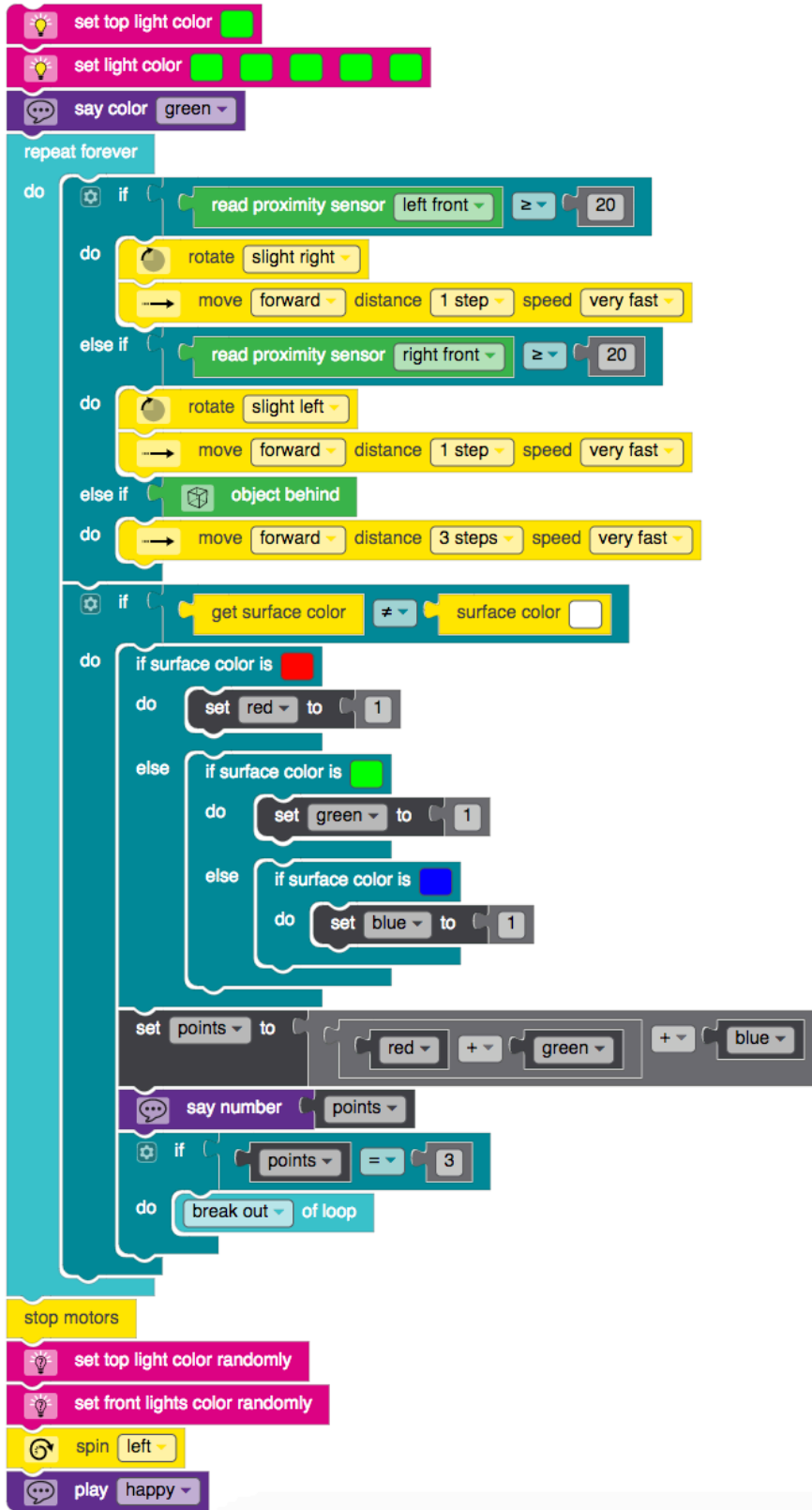
What is your creation? _____

What code will you need? (use the back of the page if needed)

Reflect

What's the hardest part of programming?
 What would you build if you could build any program for a robot or computer?

Evo's Color Quest Program



```
set top light color green
set light color green green green green green
say color green

repeat forever
do
  if (read proximity sensor left front >= 20)
  do
    rotate slight right
    move forward distance 1 step speed very fast
  else if (read proximity sensor right front >= 20)
  do
    rotate slight left
    move forward distance 1 step speed very fast
  else if (object behind)
  do
    move forward distance 3 steps speed very fast

  if (get surface color != surface color)
  do
    if surface color is red
    do
      set red to 1
    else
      if surface color is green
      do
        set green to 1
      else
        if surface color is blue
        do
          set blue to 1

    set points to (red + green + blue)
    say number points
    if (points = 3)
    do
      break out of loop

stop motors
set top light color randomly
set front lights color randomly
spin left
play happy
```

The code is a Scratch script for a robot named Evo. It starts with three initialization blocks: 'set top light color' (green), 'set light color' (five green squares), and 'say color' (green). A 'repeat forever' loop contains several conditional blocks. The first part of the loop handles proximity sensors: if the left front sensor is at least 20, rotate slightly right and move forward 1 step; if the right front sensor is at least 20, rotate slightly left and move forward 1 step; if there is an object behind, move forward 3 steps. The second part of the loop checks the surface color. If it's red, set 'red' to 1; if green, set 'green' to 1; if blue, set 'blue' to 1. Then, it calculates 'points' as the sum of red, green, and blue. It says the number of points and checks if it's 3. If so, it breaks out of the loop. After the loop, it stops motors, sets the top light color randomly, sets the front lights color randomly, spins left, and plays a happy sound.

