OZOBLOCKLY SKILLS 4
MOVE STRAIGHT

ESSENTIAL QUESTION
How can we make Ozobot Bit or Evo go perfectly straight when moving forward/ backward without a line?

OVERVIEW
Have you noticed that Ozobot may not go completely straight when you are using the "move forward/backward" command? This is due to hardware … Every Ozobot is a bit different – some swerve a little to the left or right. But with a little bit of programming help your Ozobot will go straight and follow the road!

Your students will learn how to program Ozobot Bit and Evo using OzoBlockly so that Ozobot goes straight. They will use blocks from the advanced mode, and can optionally create a function that can be used instead of "move forward/ backward."

Since students are unlikely to receive the same Ozobot day-to-day, and accuracy changes slightly over time, encourage students to learn how to change wheel bias as a skill. They don't need to memorize the wheel bias they've found today.

LESSON OUTLINE

1. Students test how Bit and Evo move forward with the "move forward" command, and see how far their Ozobot deviates from straight.
2. Students learn how to use the "set wheel speeds" to bias the wheel movement in order to make Ozobot go perfectly straight.
3. With this knowledge, students use their new skill to design a program
   a. Optionally, students can discover how to make a function out of their program. This function can then be *called* instead of using the OzoBlockly "move" command when it is essential that Ozobot goes straight.
   It's recommended that students start to use functions, since practice helps develop understanding quickly.

PREREQUISITES
Knowledge of how to program Ozobot Bit and Evo using blocks from the Movement category. To practice, go through OzoBlockly Skills 1 and 2. For groups of students, complete Lesson 3 Pair Programming beforehand.

GROUPING
Groups of two or three students

GRADE LEVEL
Grades 2 and up

MATERIALS
- Tablet or computer with OzoBlockly editor ozoblockly.com/editor
- Ozobot Bit or Evo, one per group
- Printouts of the Test Track, one per group
- Pencil to mark Ozobot's path

OZOBLOCKLY PROGRAMMING TOPICS
Free Movement, Functions (optional)

OZOBLOCKLY MODE
Modes 3 and up

DURATION
30 minutes, 50 with extension

VOCABULARY
- *Ozobot Bit or Evo* - Little robot that can follow drawn lines or can be programmed using visual codes or through the OzoBlockly programming language
- *OzoBlockly* - A visual editor which allows you to create programs by plugging blocks together. The blocks can be used to control Ozobot's behavior like movement, LED lights, etc.
- *Pair Programming* - two programmers work together, one as the Driver (at the computer) and the other as the Navigator (giving ideas and directing the course).
- *Steps* – a unit of movement forward
- *Free Movement* – Ozobot's programmed movements from OzoBlockly that do not use line following.
- *Function* - a computer programming concept wherein a code that is needed often is saved under its own name to be used any time.
- *Function Call* – the code block that represents the function and is used in the main program.

QUESTIONS ABOUT THIS LESSON?
Please contact us at ozoEdu@ozobot.com

LESSON

PART 1: TEST MOVE FORWARD

When you see this,

have your pair programmers switch Pair Programming roles.

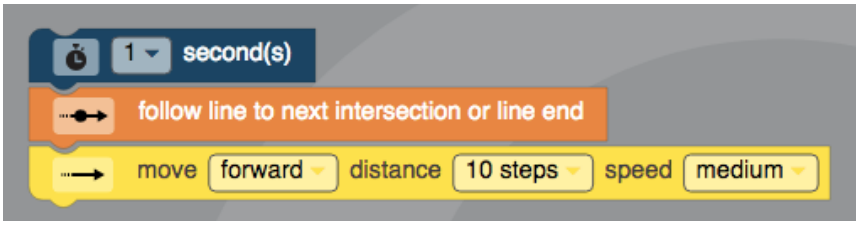
Begin by placing students into **Pair Programming** groups or pairs.

Students open OzoBlockly.com and explore Modes 3 and 4, and categories Movement and Line Direction. How many different ways can students make Ozobot move forward?

Next, hand out the track printout to your students. Point out that on the track there is a black line that Ozobot loves to travel on, but then a dotted line it won't be able to follow. This thick black line is here to help Ozobot align straight before moving into Free Movement.

Either give the correct solution or ask students for input on how they could build a launch from the line and out into open space with the codes they've explored. Their goal is to see whether and how far their Ozobot strays from the center straight line.

Write the correct answer on the board (or show them with your computer and/or projector) which blocks they'll need: under **Line Movement** use **"follow line until the next intersection or line end"** followed by **"move forward 10 steps at medium"**.



The initial time delay allows you to adjust Ozobot on the map

This code tests the **"move forward"** action from Mode 3. (All students should keep medium speed.) Each group can write the code then upload it to their Ozobot. Double-press the power button to start the code on the bottom of the black line. What result does each group get? Test several times and see if the deviation is always the same.

Students use their pencils to record the location that their Ozobot lands the most at the end of this code. Mark that location with a note like "move forward 10".

Students have now discovered that the straight movement in the lower modes of OzoBlockly isn't perfectly straight. Ozobot Evo has higher accuracy than Bit, so if you're using an Evo you might find that adjustments are small or unnecessary.

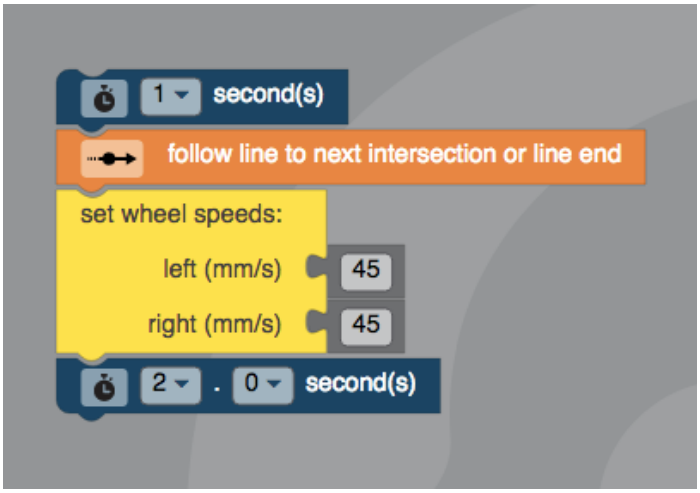
ROLES SWITCH

PART 2: SET WHEEL SPEEDS

What did your students see in the code blocks they explored that could be an alternative to Move Forward?

Either allow students to test some ideas, or show them the answer: "**set wheel speeds**" in Mode 4.

Now test the code as above, but replace **"move forward 10 steps"** with **"set wheel speeds"**, followed by a timer for **2 or 3 seconds**. It's suggested that the students work with 45 left and 45 right initially as the Ozobots tend to be straighter at that speed, and it gives greater range of tweaking.



To get an idea of how they should tweak their wheel speeds, discuss or show a video of a car taking a sharp turn. They should notice that the wheel on the inside (closest to the corner of the turn) actually slows down while the wheel on the outside of the turn has to be faster to account for the loss of movement from the other side. Your students should be able to understand how that example relates to their Ozobot.

All students can now adjust wheel speeds.

They should record with a pencil the results as they go, until they get the best answer. They should write a note with the mark of the straightest path with which wheel speeds they chose (so 43/45 or Left 43 Right 45).

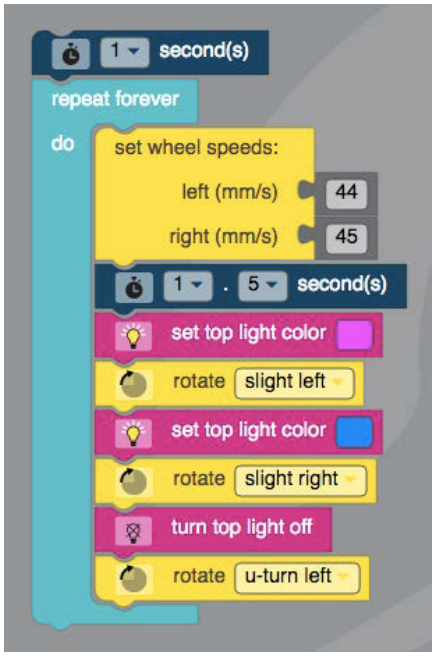
Class Responds: What is your Ozobot's preferred wheel speeds?

Are your students ready to learn functions? If so, skip Part 3 and go directly to the Extension below.

PART 3: USE THE NEW METHOD

Time to play! Now we can use our new forward to go straight with additional codes. Students code their new wheel speeds with turns and a creative dance move. Place a repeat forever code block around the movements. Use this as a guide:

- Wait 1 second
- Forever loop around:
    - Go straight
    - Creative dance move and lights
    - 90 degree turn or U-turn



These dances can be individually assessed by the teacher, or presented by the students to others. Or, students could take videos of their dances to send home.

Students who add in turns may be disappointed that the turns aren't accurate. That's okay, we're going to fix that next!

EXTENSION

Functions are a fundamental part of coding. If your students are ready to learn what a function is, then use this extension starting from the end of **PART 2**.

This extension is intended for classes where the first part is completed quickly. Functions are formally introduced in Lesson 6.

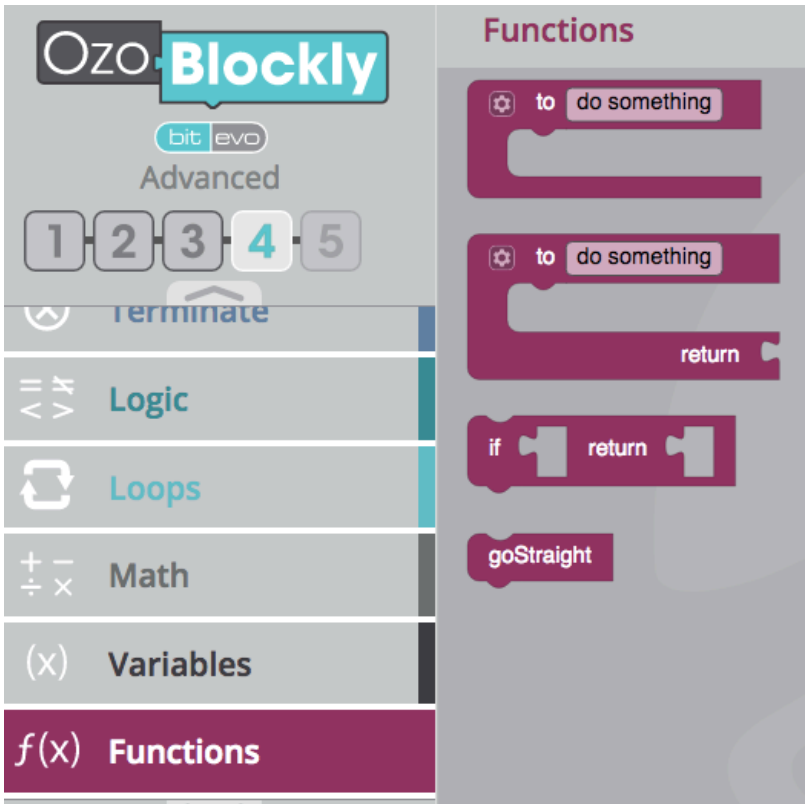
ROLES SWITCH

DEMONSTRATION: WHAT IS A FUNCTION?

Now that we have a special way of moving forward, we should save this somehow and be able to use it at any time. That's where functions come in.

A function is a few or many lines of code (blocks in this case) that are saved under a special name, and can be used in the main code block whenever we want it. We simply have to "call" the function in order to get that saved action to run. We can use that same action once or many times in the program.

Demonstrate to students: to find **Functions**, go to Mode 4. The bottom of the categories is "f(x) Functions." We just need the first block: "to (do something)".



You can change the name of that 'do something' to be anything. If your students are interested, you could show them *camelCase*.

> CamelCase is a way that programmers save names for files, functions etc. In programming languages, spaces aren't allowed in function names, but sometimes the names need to be descriptive. That's why we use camelCase - the first word can be lower-case, but each following word is capitalized (yes, even those tiny prepositions and articles), e.g., goStraight, makeAUTurn, twoMinuteDance.

Make a function by putting the blocks you want inside the magenta 'do something' block (like in example below). To use the function, use the newly created '**goStraight'** Function Call in the **Functions** category, which appears in image above. You can see '**goStraight**' is already there. That's because the function had been created already.

Example program with **goStraight** function:

The function runs in the main program on the left because the Function Call is in the main program. The function alone (right) won't run, even if it's on the screen, unless it's "called" inside the main program you're writing (left).

Attach "**goStraight**" from the **Functions** section onto the main program, as shown in the example above. You can use 'goStraight' as many times as you want.

Class Responds: Why would we make a function, instead of just putting all the code blocks together? Possible answers are that it:

- shortens the actual running code,
- makes it easier to read for human eyes,
- is easier to edit,
- reduces bugs,
- keeps the programmer more organized,
- makes the program run more efficiently on the computer, etc.

PART 4 USE THE NEW FUNCTION

ROLES SWITCH

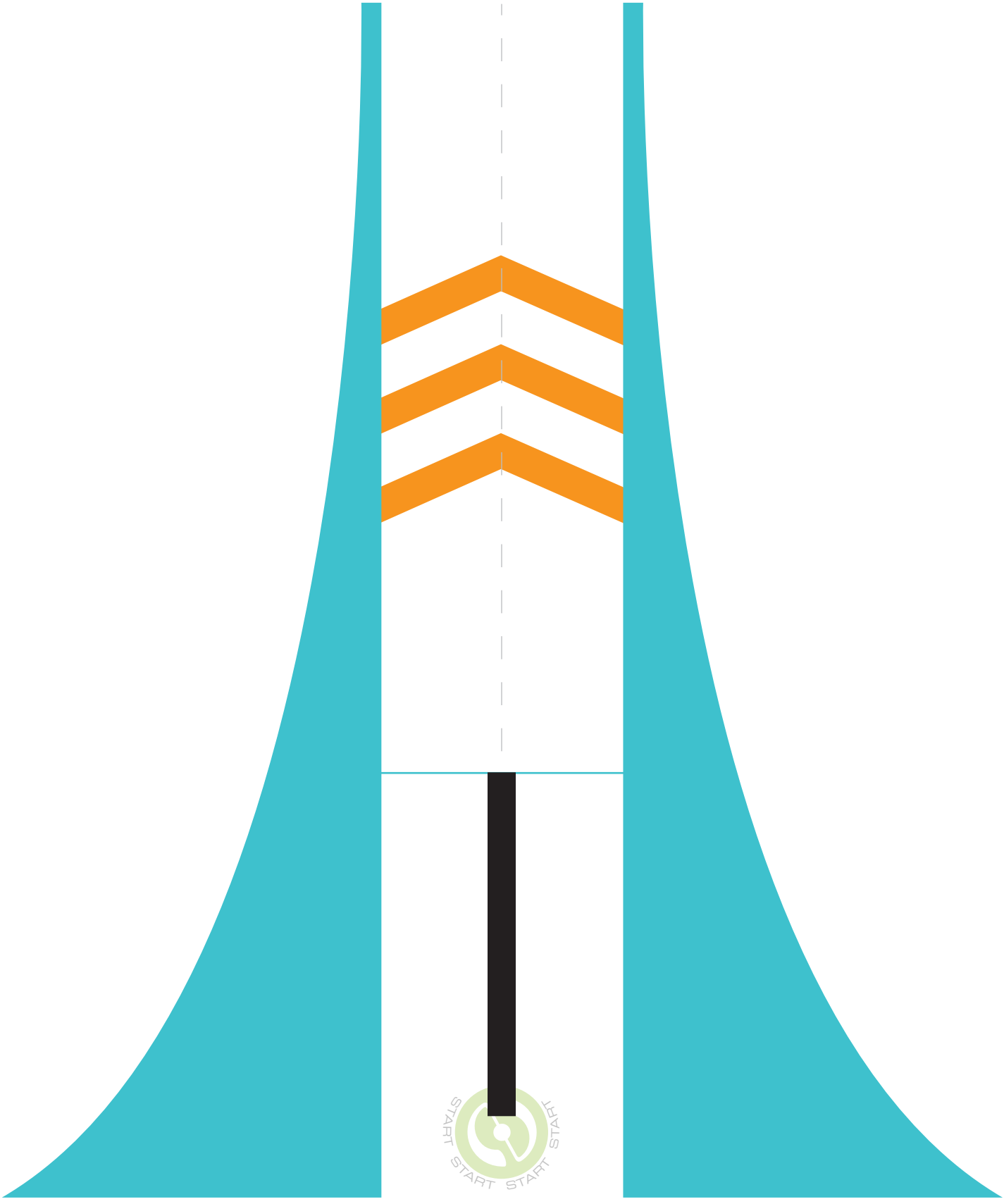
Time to play! Now we can use our function to go straight a few times. Students code their **goStraight** function (as above) and a special move. Students can add the following:

- fun moves in between **goStraight**
- spins between **goStraight**
- turns to complete a square

Example: Ozobot draws a square

Example: Ozobot does Travolta's Point Move dance from Saturday Night Fever
top left: the code block that's actually running; top right, bottom right: functions that only run when you "call" the code "discoMoveTravolta" and "goStraight"

Students who add in turns may be disappointed that the turns aren't accurate. That's okay, we're going to fix that next!



ozobot TEST TRACK