Evo Lap Counter

By Richard Born
Associate Professor Emeritus
Northern Illinois University
rborn2@niu.edu

Introduction

Imagine you are walking laps in the local gym. Maybe you are with a friend and are chit chatting during the walk. Or maybe you are just by yourself listening to music on your headset while your mind wonders. You suddenly realize that you have lost track of the number of laps you have done. Wouldn't it be nice to have a lap counter that you could "click" each time you finish a lap?

In this lesson, you are going to turn Evo into a lap counter. You will be creating an OzoBlockly program to accomplish this. Each time that Evo's button is pressed, Evo will speak the number of the lap just completed. Your program will make use of blocks all the way through some from Master Mode 5. Blocks will be used from the following categories: light effects, sounds, button, timing, loops, logic, and math.

When creating a program, it is always advisable to know the program requirements. Here are the lap counter specifications to keep in mind as you create the counter:

1. *Evo should display a red top light for two seconds when the program starts.*
2. *The lop light should then turn green, letting the user know that Evo is ready to start counting laps.*
3. *The green light should be very dim in order to conserve Evo's battery while the user is walking.*
4. *Each time the user presses Evo's button, Evo should speak the lap number, starting with 1, then 2 on the next press, etc.*
5. *After each press of the button, Evo's top light should also produce a quick flash of white light. This would be helpful for a user who is hearing impaired just to make sure that Evo recognized the button press.*
6. *If the lap count should ever reach 127, the program should terminate and turn Evo off after speaking the lap number 127.*

With these specification in mind, advanced computer science students could be challenged to create the program from scratch, *i.e.*, without any further instructions or help.

Students with less knowledge of OzoBlockly programming would probably be better off to build the program while following the detailed discussion on the remaining pages of this document.

As a side note, the author of this lesson uses the Evo Lap Counter every day during his walk on the "11 laps per mile" track at a local fitness center. Evo is simply held in his hand and clicked after finishing each lap. This gives the author more time to think about new Ozobot lessons and not trying to remember the lap number!

Creating the Evo Lap Counter

Let's walk through a solution to create the Evo Lap Counter. You should open OzoBlocky and build the program in steps as described here.

Take a look at Figure 1 for the first blocks in our program. It begins with some housekeeping. The "capture button press events to true" block allows Evo to accept button presses from the user. Without this block, the default value of false would turn Evo off if the user pressed the button. The "set button press count to 0" block does just what you would expect—it initializes the value of a special built-in button press counter to the value zero. The "set top light color" and "wait" blocks accomplish requirement 1 from the requirements list on the previous page of this document.

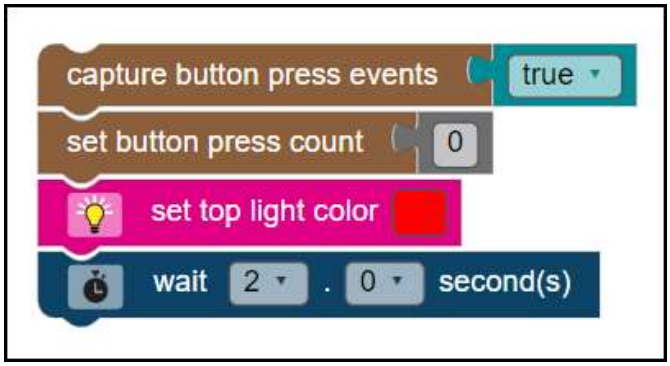

Figure 1

As shown in Figure 2, we next add a "repeat while true" do loop block, sometimes referred to as a *forever loop*. We'll see later in this discussion how to get out of a forever loop. Most of the rest of our program will consist of blocks that we insert into this loop.

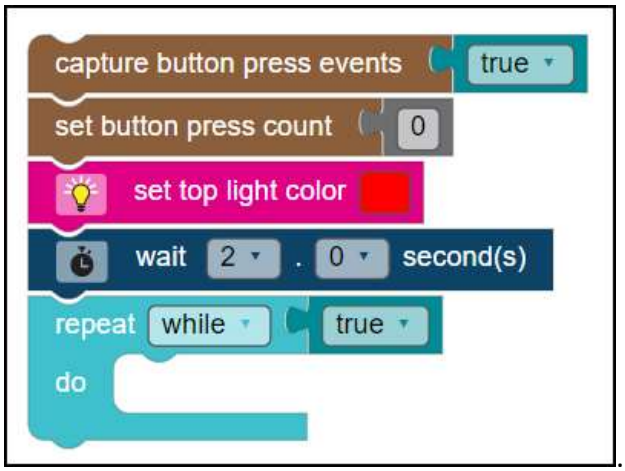

Figure 2

Figure 3 shows that we have inserted a "set top light color" block into the forever loop. As per requirement 2, the lop light is set to green, letting the user know that Evo is ready to start counting laps. The number that appears for the green operand can be any number between 0 and 127, with 0 meaning

no green and 127 meaning the brightest possible green component. A value of 5 makes the green very dim in order to conserve Evo's battery while the user is walking laps.

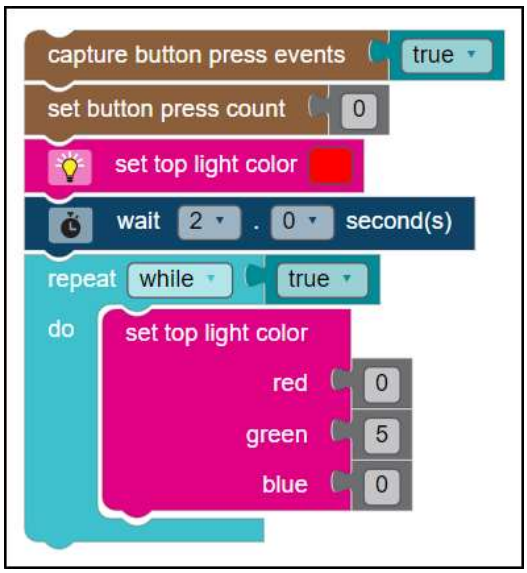

Figure 3

As shown in Figure 4, two additional blocks are added to the forever loop. First, there is a "wait for button press" block. Control of the program stays in this block until the user presses Evo's button. As soon as the button is pressed, the button press count is automatically increased by 1. Then control passes to the "say number" block, where Evo speaks the current button press count using "get button press count". This satisfies requirement 4: "*Each time the user presses Evo's button, Evo should speak the lap number, starting with 1, then 2 on the next press, etc.*"

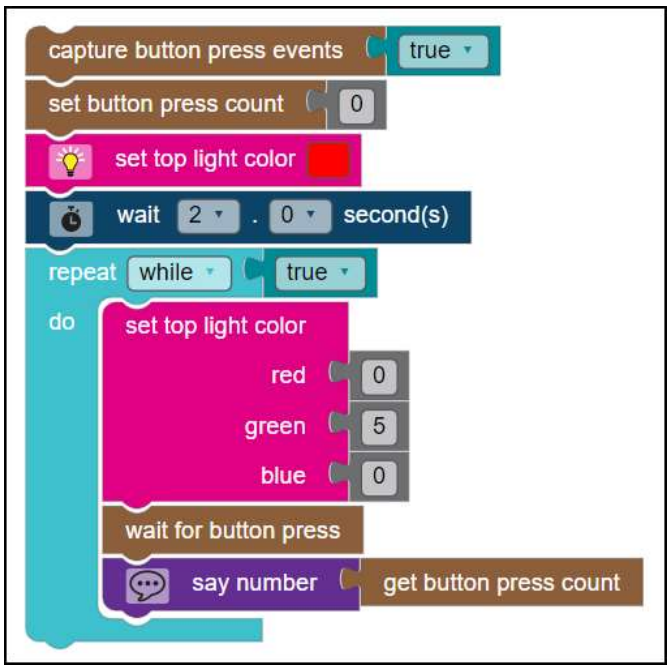

Figure 4

Figure 5 shows four added blocks that satisfy requirement number 5, causing Evo to quickly flash its top light white. The top light is set to white, there is a 50 ms pause, it is turned off, and there is another 50 ms pause. Note that 1 ms = 1/1000 s. Therefore, each pause is 50/1000 = 1/20 s. We could have used the block 

, but then the minimum wait time would be 0.1 s =1/10 s. You can actually use whichever you find most appealing and adjust the parameters to your liking.

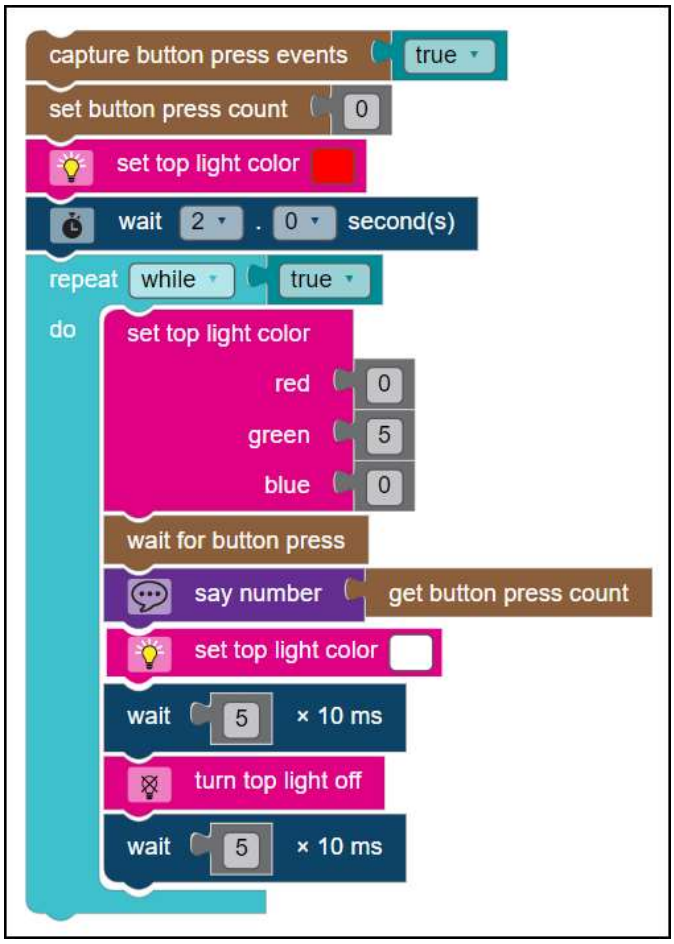

Figure 5

As shown in Figure 6, we have added one more block to the forever loop. This "if…do" block checks to see if the current button press count is 127. If so, control exits the loop with a "break out of loop" block, and control transfers to the "terminate program and turn Ozobot off" block. Note that the purpose of using the value 127 is that 127 is the largest possible positive integer allowed by OzoBlockly. If we had not included this "if…do" in our program, then the program would have an abnormal termination when the button press count reached a value of 128. Evo's top light would then alternately flash red and blue to indicate failure.

When the condition in the "if..do" is false, then the loop would be repeated again, and Evo's top light would again be set to green.

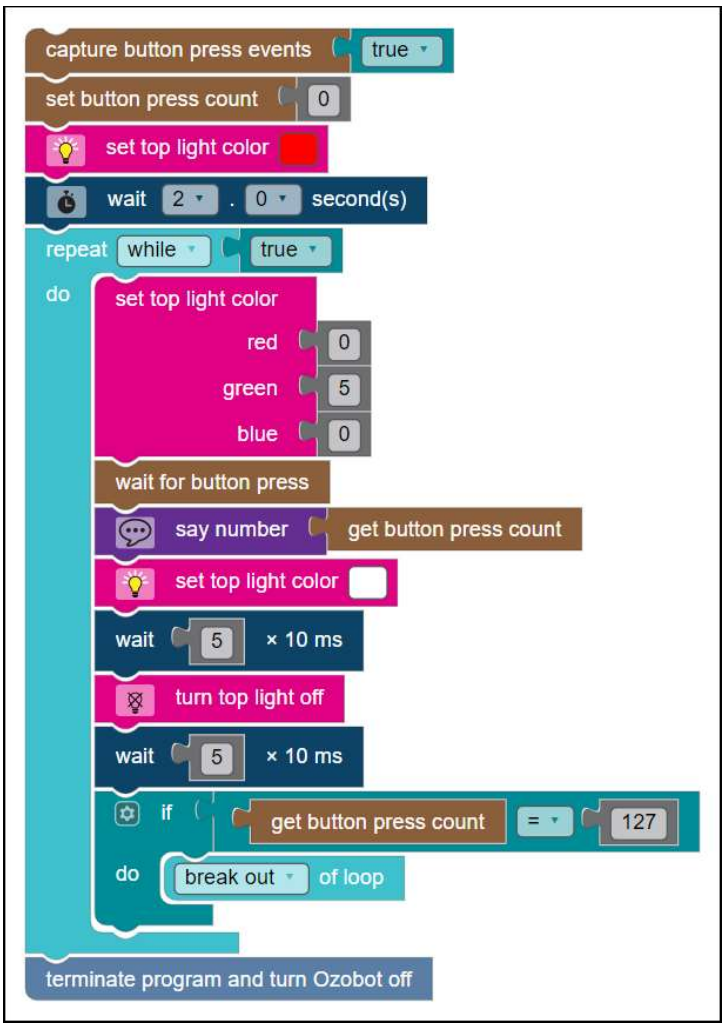

Figure 6

Now that you have created the entire program in OzoBlockly, it is time to load the program into Evo and then run the program. If you have made no errors in program development, the program should run according to all of the program requirements!

You can, of course, use this Evo counter to count anything you want. Maybe your school is having some kind of a party and you want to count guests as they arrive. Or maybe you want to count donations for a food drive. Or maybe … just about anything countable—provided that the count is not expected to exceed 127!