Unit 2: For Loops | **Skill Builder 2: Loop through COLORs**

In this second lesson for Unit 2, you will learn about controlling the intensity of the three colors of the COLOR LED on the TI-Innovator™ Hub.

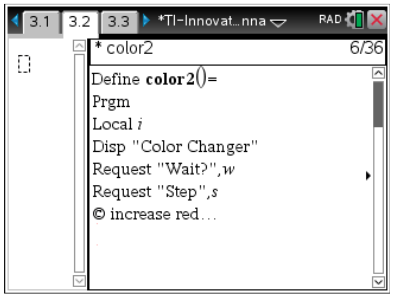
Objectives:

- Use **For** loops to control each of the three color channels on the COLOR LED

The red, green, and blue values (from 0 to 255) sent to the COLOR LED determine the brightness of each color channel. This program demonstrates varying the amount of each color gradually to transition through some of the over 16 million (256^3) colors possible. You will again use some **For** loops in your program.

Creating a Color Changer program

1. Start a new program, and call it COLOR2.
2. Add **Disp** with the title of the program as shown.
3. Add **Request** and, in quotations marks, add the text *Wait?*.
4. Then add a comma and the variable ***w***.
   - This variable will be used in a **Wait** statement, so the lower the number, the lower the wait time and the faster the program executes the next command.
5. As shown on the right, also request a **STEP** value to be used in the ***For*** loop to speed things up a bit.

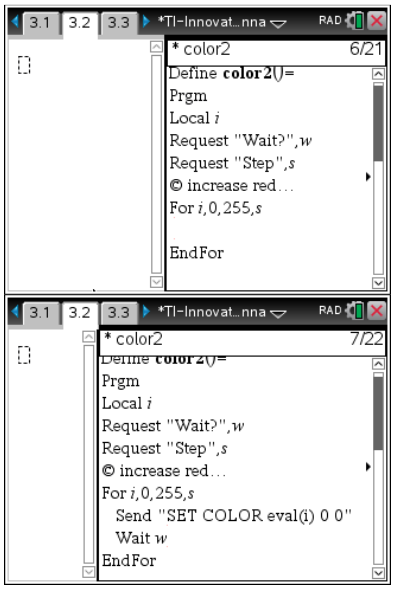


Notes: **Local i** *prevents the variable from being created in the problem (outside the program). The comment symbol © is available from* ***menu > Actions > Insert Comment***.

Our program will gradually (depending on the wait and step values) increase the RED intensity, then add GREEN, then gradually take away the RED, then add BLUE, take away GREEN, then add RED to the BLUE, then take away the BLUE, then finally take away the RED. This is a rather long program, and you can run it after you complete each of the For loops to test things out.

It's a good thing that the editor automatically provides both the **For** statement and its corresponding **EndFor** statement *at the same time* so that you don't forget it later on.

6. Add a **For…EndFor** loop (from the Control menu) after the two **Request** statements.

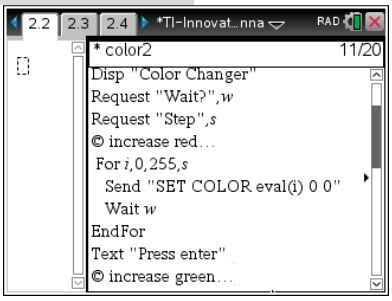


Complete the First Loop

7. Add the rest of the pieces of the **For** statement to make the value of *i* increment from 0 to 255. Use the loop variable ***i*** and the step variable ***s***.
8. Add the **Send "SET COLOR** statement from the prgm **HUB** menu.
9. Use the **eval(**) function from the **HUB** menu for the variable ***i*** to control the red channel and set the GREEN and BLUE channels to 0.
10. Remember to close the quotes and the parentheses.
11. Follow the **Send** statement with the **Wait** statement using the variable ***w*** that you used in the **Request** statement earlier.

Teacher Tip: After completing this first loop, students can run the program to see the COLOR LED gradually brighten to RED. Use a small Wait value, like 0.1 and a large step value, like 10.

12. After the End of this first **For** loop, you can use the **Text** statement with a message just so you can admire that bright red LED.



Add the Green Light Loop

Now we'll build another **For** loop to add GREEN to the LED. But this time we want to only control the GREEN channel and not touch the RED channel. We can do this in two ways:

Send "SET COLOR 255 eval(i) 0"
(since we know that the RED is all the way on and the BLUE is off)

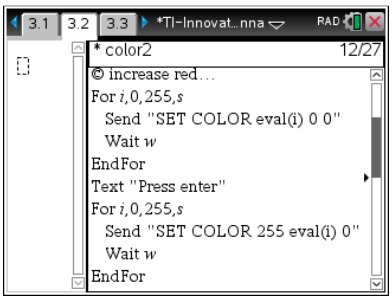
or

Send "SET COLOR.GREEN eval(i)"

This second statement controls only the GREEN channel and does not affect the RED and BLUE channels. In both cases, notice that we can re-use the variable **i** from the first **For** loop.

In the image to the right, note that we chose to use the former method.

1. Add the **Wait** statement inside the loop body using the variable ***w***.
2. Add the **Text** statement *after* the **End** of the loop again to admire the new color. What color is that?



Now we want to gradually decrease the amount of RED so that we are left with only GREEN.

To *decrease* in a **For** loop, we start with the highest number, go to the lowest number, and use a *negative* step value:

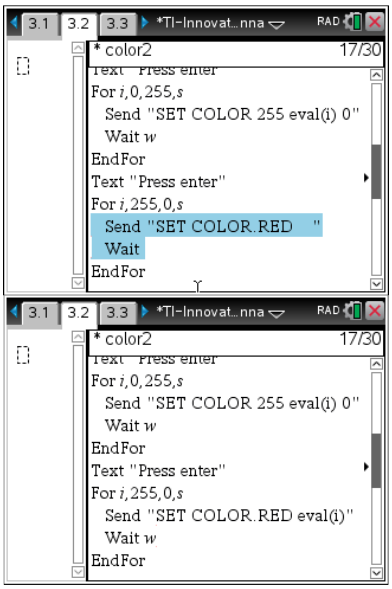
For i, 255, 0, -S

starts at 255 and subtracts S in each step of the loop until the variable **i** is less than 0 when the loop ends. Be sure to use the (-) 'negative' key and not the subtract key. That would cause an error.

We only want to change the RED channel so we'll use COLOR.RED in the **Send** statement.

The rest of this loop is similar to the first two loops we constructed. The image to the right shows only the *keywords* entered.

Can you complete each of these statements? If not, refer to the next step.



Here's the completed section that removes the RED gradually. At the end of this loop, you should see a bright GREEN color:

3. Now add a loop to add BLUE.
4. Then add a loop to remove GREEN.
5. Then add a loop to add RED again.
   - What color do you see at the end of these loops?
6. Then add a loop to remove BLUE.
7. Finally, add a loop to remove RED.
   - What color is the LED at the end of the program?
   - What happens when all three color channels are 0?

Teacher Tip: When RED and BLUE are on we get purple (magenta). When BLUE and GREEN are on, we get Cyan. At the end of the program, the LED might not be off. That's because the last color values sent to the LED might not be 0. To be sure the LED is off at the end of a program, send "SET COLOR 0 0 0".