

Unit 4: Driving Features

Skill Builder 2: Light at the Corners

In this lesson, you will learn about two interesting features of Rover: the color LED and waiting for Rover to be ready for the next task.

Objectives:

- Control Rover's color LED
- Wait for Rover to finish the current task before starting a new task
- Lighting up the LED at the proper time

On top of Rover near the battery strength meter (the 4 green lights) there is a color LED just like on the TI-Innovator™ Hub. But you cannot see the TI-Innovator Hub's LEDs. This LED works like the TI-Innovator Hub LED. It requires values in the form (red, green, blue).

In a new Rover Coding template program, try the command found on

menu > TI Rover > Outputs > color_rgb()

which is pasted into your program as

rv.color_rgb(red, green, blue).

Provide values for the three color channels (0 to 255 each) and run the program to see the LED light up in your color.

1. In the previous lesson, you made Rover drive in a square pattern using a **for** loop. In this lesson, you will have the color LED light up in red *at the corners only*.

2. Make a *copy* of your square program from Unit 4 Skill Builder 1.

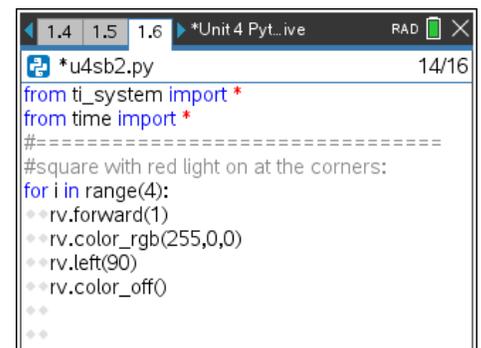
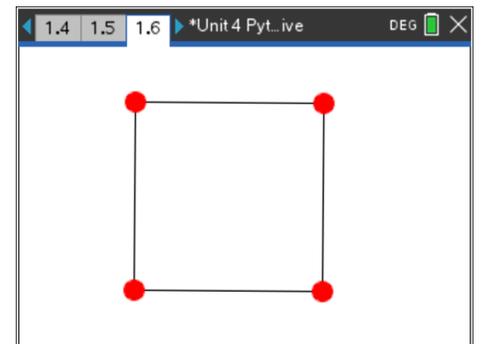
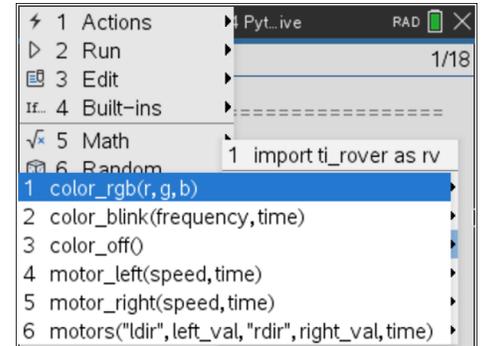
menu > Actions > Create copy...

Just before the turn statement (left or right) in your program, turn the LED red. After the turn statement, turn the LED off:

rv.color_rgb(255,0,0)
rv.left(90)
rv.color_off()

Both color statements are on **menu > TI Rover > Outputs**.

Run the program. Does it do what you expected?





- This attempt does not work correctly because the TI-Nspire CX II is working faster than the Rover. The TI-Nspire CX II sends all the instructions to the TI-Innovator Hub as fast as it can. The TI-Innovator Hub then stores the *driving* instructions and processes them as a driver following a route on a mapping app, one at a time. The drive commands are stored in a 'queue' (a list) and are processed one at a time because each instruction takes some time to finish.

But the `rv.color` command is *not* a driving command. When the TI-Innovator Hub receives this instruction, it is processed *immediately* (independent of the driving commands). So, the LED blinks rapidly four times right at the start of the driving.

- Fortunately, there is a statement that you can use to tell the program to wait until Rover is 'ready' to turn. After the `rv.forward()` statement add the statement

`rv.wait_until_done()`

found on **menu > TI Rover > Commands**.

This instruction is telling the TI-Nspire CX II to wait until it receives a signal from Rover that the Rover is finished with the `rv.forward ()` command. The red LED then comes on and Rover turns.

Test your program now.

- Notice that the LED flashes quickly at the beginning of the turn. The turning also takes some time to complete, so you must tell the TI-Nspire CX II to wait again while the turn is taking place before turning the LED off.

Add another `rv.wait_until_done()` statement *after* the `rv.left()` statement and before the LED is turned off so that the LED stays on *throughout* the entire turn.

Try your program again.

Can you have the LED light up in another color along the sides of the square? Hint: It only takes one more statement.

Drive Queue and Other

	A	B	C	D
1	drive	other		
2	forward 1	LED red		
3	left 90	LED off		
4	forward 1	LED red		
5	left 90	LED off		

```

#square with red light on at the corners:
for i in range(4):
  rv.forward(1)
  rv.wait_until_done()
  rv.color_rgb(255,0,0)
  rv.left()

```

```

#square with red light on at the corners:
for i in range(4):
  rv.forward(1)
  rv.wait_until_done()
  rv.color_rgb(255,0,0)
  rv.left()
  rv.wait_until_done()
  rv.color_rgb(0,0,0)

```