<http://www.ev3dev.org/downloads/>

<http://www.ev3dev.org/docs/getting-started/>

<https://etcher.io/>

### LEDs

|  |
| --- |
| **UPDATE SEPTEMBER 2017:** Although controlling the LEDs (plural, since EV3 Python controls the left and right LEDs separately) should be simple and straightforward, I have actually experienced significant frustration with this. My experiments lead me to think that errors may occur if an LED operation does not have time to complete before another is attempted, and I also get the impression that trying to make the LEDs flash can often give unexpected results, especially for the colors orange, amber and yellow. These are 'composite' colors made by blending the red and green colors which are the colors of the actual LEDs. If you try to flash red and green you are more likely to be successful. If you experience problems then my advice is to include **sleep(0.01)** after each LED command and avoid trying to make the LEDs flash. Here is an example of code that follows my advice- it sets both LEDs to orange:  from ev3dev.ev3 import \*  from time import sleep  Leds.set\_color(Leds.LEFT,  Leds.ORANGE)  sleep(0.01)  Leds.set\_color(Leds.RIGHT, Leds.ORANGE)  sleep(0.01)  However, since I am not sure that other people are experiencing these problems I will assume that this is *not*the case for the rest of this page.  -----------------------------------------------------------------------------------------------------------------------  Controlling the two LEDs located under the buttons of the EV3 brick must be one of the simplest things we can do in EV3 Python so let's start with that.  The official documentation is [**HERE**](http://python-ev3dev.readthedocs.io/en/latest/other.html#leds). Note that the Leds class is a child of the Led class so you should perhaps look at the documentation for that too.  I think that all you really need to know as a beginner is how to set the color of each LED and how to make the LEDs blink. For other functions such as setting the brightness of each LED, see the official documentation.  **Set the color of an LED**  Use Leds.set\_color(*group*, *color*)  Group can be Leds.LEFT or Leds.RIGHT.  Color can be Leds.RED, Leds.GREEN, Leds.YELLOW, Leds.ORANGE orLeds.AMBER.  **Example**  This example turns the left led red when the touch sensor button is pressed. When you run this or any other program, be aware that the program will take a few seconds to start. Press Ctrl-C to interrupt the infinite loop.  #!/usr/bin/env python3  # Above line is needed so program can be run from Brickman  # Plug a touch sensor into any sensor port  from ev3dev.ev3 import \*  ts = TouchSensor()  while True:      if ts.value()==1:   #touch sensor pressed          Leds.set\_color(Leds.LEFT, Leds.RED)      else:          Leds.set\_color(Leds.LEFT, Leds.GREEN)  # Press Ctrl-C to exit  Here is the same example in a neater but more obscure form. Highlighted is a tuple with two elements. If ts.value() is zero then element zero of the tuple (green) will be used, etc.  #!/usr/bin/env python3  # So program can be run from Brickman  # Plug a touch sensor into any sensor port  from ev3dev.ev3 import \*  ts = TouchSensor()  while True:      Leds.set\_color(Leds.LEFT, (Leds.GREEN, Leds.RED)[ts.value()])  # Press Ctrl-C to exit  Here is a similar example which will switch both LEDs at the same time:  Coming...  **Make the LEDs blink**  Here’s an LED blink demo. **Warning:** this currently does not work correctly for me if I choose colors other then RED or GREEN which are the true colors of the LEDs. The other colors, ORANGE, YELLOW and AMBER, are made by using the RED and GREEN LEDs together, and for me this sometimes causes the wrong color to appear or error messages to be triggered.    #!/usr/bin/env python3  from ev3dev.ev3 import \*  Leds.set(Leds.LEFT, brightness\_pct=0.5, trigger='timer')  Leds.set(Leds.LEFT, delay\_on=3000, delay\_off=500)  Leds.set\_color(Leds.LEFT, Leds.GREEN)    This example uses the “Leds.set” method to set some properties on the left (“LEFT”) LED group. It sets the brightness to 50%, then it sets the trigger to “timer”, which means that it will be “triggered” on a set timer. “delay\_off” and “delay\_on” dictate the amount of time (in milliseconds) that the LED is off or on, respectively. With the numbers given here, it will stay off for 3 seconds and then turn on for half a second, then repeat. The trigger must be set to 'timer' *before*delay\_off and delay\_on are set, and in practice this means they must be set within separate commands. It's possible that when the left and right LEDs blink they may not be synchronised.  Note that the settings will keep their values after the script stops running, so the LEDs will continue to blink until some other script stops them.  **Turn off the LEDs**  Turn off both LEDs with Leds.all\_off() |

LEDs

UPDATE SEPTEMBER 2017: Although controlling the LEDs (plural, since EV3 Python controls the left and right LEDs separately) should be simple and straightforward, I have actually experienced significant frustration with this. My experiments lead me to think that errors may occur if an LED operation does not have time to complete before another is attempted, and I also get the impression that trying to make the LEDs flash can often give unexpected results, especially for the colors orange, amber and yellow. These are 'composite' colors made by blending the red and green colors which are the colors of the actual LEDs. If you try to flash red and green you are more likely to be successful. If you experience problems then my advice is to include sleep(0.01) after each LED command and avoid trying to make the LEDs flash. Here is an example of code that follows my advice- it sets both LEDs to orange:

from ev3dev.ev3 import \*

from time import sleep

Leds.set\_color(Leds.LEFT, Leds.ORANGE)

sleep(0.01)

Leds.set\_color(Leds.RIGHT, Leds.ORANGE)

sleep(0.01)

However, since I am not sure that other people are experiencing these problems I will assume that this is not the case for the rest of this page.

-----------------------------------------------------------------------------------------------------------------------

Controlling the two LEDs located under the buttons of the EV3 brick must be one of the simplest things we can do in EV3 Python so let's start with that.

The official documentation is HERE. Note that the Leds class is a child of the Led class so you should perhaps look at the documentation for that too.

I think that all you really need to know as a beginner is how to set the color of each LED and how to make the LEDs blink. For other functions such as setting the brightness of each LED, see the official documentation.

Set the color of an LED

Use Leds.set\_color(group, color)

Group can be Leds.LEFT or Leds.RIGHT.

Color can be Leds.RED, Leds.GREEN, Leds.YELLOW, Leds.ORANGE or Leds.AMBER.

Example

This example turns the left led red when the touch sensor button is pressed. When you run this or any other program, be aware that the program will take a few seconds to start. Press Ctrl-C to interrupt the infinite loop.

#!/usr/bin/env python3

# Above line is needed so program can be run from Brickman

# Plug a touch sensor into any sensor port

from ev3dev.ev3 import \*

ts = TouchSensor()

while True:

if ts.value()==1: #touch sensor pressed

Leds.set\_color(Leds.LEFT, Leds.RED)

else:

Leds.set\_color(Leds.LEFT, Leds.GREEN)

# Press Ctrl-C to exit

Here is the same example in a neater but more obscure form. Highlighted is a tuple with two elements. If ts.value() is zero then element zero of the tuple (green) will be used, etc.

#!/usr/bin/env python3

# So program can be run from Brickman

# Plug a touch sensor into any sensor port

from ev3dev.ev3 import \*

ts = TouchSensor()

while True:

Leds.set\_color(Leds.LEFT, (Leds.GREEN, Leds.RED)[ts.value()])

# Press Ctrl-C to exit

Here is a similar example which will switch both LEDs at the same time:

Coming...

Make the LEDs blink

Here’s an LED blink demo. Warning: this currently does not work correctly for me if I choose colors other then RED or GREEN which are the true colors of the LEDs. The other colors, ORANGE, YELLOW and AMBER, are made by using the RED and GREEN LEDs together, and for me this sometimes causes the wrong color to appear or error messages to be triggered.

#!/usr/bin/env python3

from ev3dev.ev3 import \*

Leds.set(Leds.LEFT, brightness\_pct=0.5, trigger='timer')

Leds.set(Leds.LEFT, delay\_on=3000, delay\_off=500)

Leds.set\_color(Leds.LEFT, Leds.GREEN)

This example uses the “Leds.set” method to set some properties on the left (“LEFT”) LED group. It sets the brightness to 50%, then it sets the trigger to “timer”, which means that it will be “triggered” on a set timer. “delay\_off” and “delay\_on” dictate the amount of time (in milliseconds) that the LED is off or on, respectively. With the numbers given here, it will stay off for 3 seconds and then turn on for half a second, then repeat. The trigger must be set to 'timer' before delay\_off and delay\_on are set, and in practice this means they must be set within separate commands. It's possible that when the left and right LEDs blink they may not be synchronised.

Note that the settings will keep their values after the script stops running, so the LEDs will continue to blink until some other script stops them.

Turn off the LEDs

Turn off both LEDs with Leds.all\_off()

LED

ACTUALIZACIÓN SEPTIEMBRE 2017: aunque el control de los LED (en plural, ya que EV3 Python controla los LED izquierdo y derecho por separado) debe ser simple y directo, en realidad he experimentado una frustración significativa con esto. Mis experimentos me llevan a pensar que pueden ocurrir errores si una operación LED no tiene tiempo de completarse antes de intentar otra, y también tengo la impresión de que tratar de hacer que los LEDs destellen a menudo puede dar resultados inesperados, especialmente para los colores naranja. ámbar y amarillo. Estos son colores 'compuestos' hechos mezclando los colores rojo y verde que son los colores de los LED reales. Si tratas de parpadear en rojo y verde, es más probable que tengas éxito. Si tiene problemas, entonces mi consejo es incluir el sueño (0.01)después de cada comando de LED y evite tratar de hacer que los LED parpadeen. Aquí hay un ejemplo de código que sigue mi consejo: establece ambos LED en naranja:

de la importación ev3dev.ev3 \*

desde el tiempo de importación de sueño

Leds.set\_color (Leds.LEFT, Leds.ORANGE)

dormir (0.01)

Leds.set\_color (Leds.RIGHT, Leds.ORANGE)

dormir (0.01)

Sin embargo, dado que no estoy seguro de que otras personas experimenten estos problemas, asumiré que este no es el caso para el resto de esta página.

-------------------------------------------------- -------------------------------------------------- -------------------

Controlar los dos LED ubicados debajo de los botones del bloque EV3 debe ser una de las cosas más simples que podemos hacer en EV3 Python, así que comencemos con eso.

La documentación oficial está AQUÍ . Tenga en cuenta que la clase Leds es un elemento secundario de la clase Led , por lo que quizás también deba consultar la documentación.

Creo que todo lo que necesita saber es que un principiante es cómo establecer el color de cada LED y cómo hacer parpadear los LED. Para otras funciones como configurar el brillo de cada LED, consulte la documentación oficial.

Establecer el color de un LED

Use Leds.set\_color ( grupo , color )

El grupo puede ser Leds.LEFT o Leds.RIGHT.

El color puede ser Leds.RED, Leds.GREEN, Leds.YELLOW, Leds.ORANGE o Leds.AMBER .

Ejemplo

Este ejemplo gira el led izquierdo al rojo cuando se presiona el botón del sensor táctil. Cuando ejecute este u otro programa, tenga en cuenta que el programa tardará unos segundos en comenzar. Presione Ctrl-C para interrumpir el ciclo infinito.

#! / usr / bin / env python3

# Por encima de la línea es necesario para que el programa pueda ejecutarse desde Brickman

# Enchufe un sensor táctil en cualquier puerto del sensor

de la importación ev3dev.ev3 \*

ts = TouchSensor ()

mientras que es cierto:

si ts.value () == 1: # sensor táctil presionado

Leds.set\_color (Leds.LEFT, Leds.RED)

más:

Leds.set\_color (Leds.LEFT, Leds.GREEN)

# Presione Ctrl-C para salir

Aquí está el mismo ejemplo en una forma más nítida pero más oscura. Destacado es una tupla con dos elementos. Si ts.value () es cero, entonces se usará el elemento cero de la tupla (verde), etc.

#! / usr / bin / env python3

# Entonces el programa puede ejecutarse desde Brickman

# Enchufe un sensor táctil en cualquier puerto del sensor

de la importación ev3dev.ev3 \*

ts = TouchSensor ()

mientras que es cierto:

Leds.set\_color (Leds.LEFT, (Leds.GREEN, Leds.RED) [ts.value ()] )

# Presione Ctrl-C para salir

Aquí hay un ejemplo similar que cambiará ambos LED al mismo tiempo:

Viniendo...

Haz que los LED parpadeen

Aquí hay una demostración de LED parpadeante. Advertencia: actualmente, esto no funciona correctamente si elijo otros colores, luego ROJO o VERDE, que son los colores verdaderos de los LED. Los otros colores, NARANJA, AMARILLO y AMBAR, se crean utilizando los LED ROJO y VERDE juntos, y para mí, esto a veces hace que aparezca el color incorrecto o que se activen los mensajes de error.

#! / usr / bin / env python3

de la importación ev3dev.ev3 \*

Leds.set (Leds.LEFT, brightness\_pct = 0.5, trigger = 'timer')

Leds.set (Leds.LEFT, delay\_on = 3000, delay\_off = 500)

Leds.set\_color (Leds.LEFT, Leds.GREEN)

Este ejemplo utiliza el método "Leds.set" para establecer algunas propiedades en el grupo de LED izquierdo ("IZQUIERDO"). Establece el brillo en 50%, luego establece el disparador en "temporizador", lo que significa que se "activará" en un temporizador establecido. "Delay\_off" y "delay\_on" dictan la cantidad de tiempo (en milisegundos) que el LED está apagado o encendido, respectivamente. Con los números que figuran aquí, permanecerá apagado durante 3 segundos y luego se encenderá durante medio segundo, luego se repetirá. El disparador se debe establecer en 'temporizador' antes de que se establezcan delay\_off y delay\_on, y en la práctica esto significa que deben configurarse dentro de comandos separados. Es posible que cuando los LED izquierdo y derecho parpadean, no estén sincronizados.

Tenga en cuenta que la configuración mantendrá sus valores después de que la secuencia de comandos deje de ejecutarse, por lo que los LED continuarán parpadeando hasta que otra secuencia de comandos los detenga.

Apague los LED

Desactive ambos LED con Leds.all\_off ()