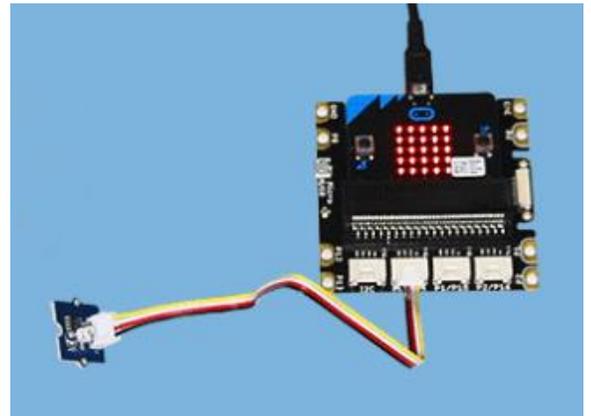


Introduction:

The micro:bit has a row of gold connections along the bottom edge for connecting electronic accessories. To make these connections easier there are many 'expansion boards' available that allow us to 'plug in' those accessories using standard cables. Two of these expansion boards are the **Grove shield** and the **Bitmaker expansion board**. The projects presented here will work with these and many other expansion boards. In addition to the expansion board, you will use some Grove accessories: an external LED (any *one* color), a light sensor, an ultrasonic ranger, and an external speaker.

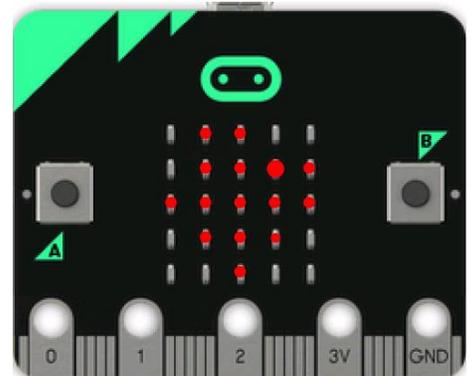
These activities assume you have completed the *first* set of micro:bit activities.

1. Before you begin, be sure that:
 - your **micro:bit** is connected to your TI-84 Plus CE Python
 - your **micro:bit** is inserted in the **expansion board**.
(a *Grove shield* is shown here)
 - for this activity, you must have a Grove LED accessory and 4-wire connecting cable as shown here connected to pin0.



2. Your micro:bit should look like this when it has power from the TI-84:
Recall that the display on the **micro:bit** is showing the TI logo, an icon of the state of Texas with a bright spot near Dallas, the home of **Texas Instruments, Inc.**

This activity has two parts: an on/off switch and a temporary-on switch. There's also a 'Challenge' at the end.

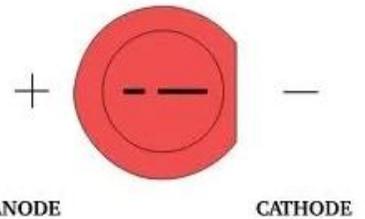


10 Minutes of Code: Python Modules

MICRO:BIT: THE LIGHT SWITCH

TI-84 PLUS CE PYTHON

- The Grove LED is a small circuit board with a white (or colored) LED on it. **Turn the yellow dial (a potentiometer) all the way to the right to get the maximum brightness.** Connect this LED board to the port labeled **P0 (pin0)** on the expansion board using the cable included.
 - Caution!** The LED can be removed from its socket on the circuit board. Be careful when re-inserting the wires properly. There is a positive (anode) wire and a negative (cathode) wire and the circuit board is labeled with '+' and '-' symbols. The negative wire on the LED is indicated by a 'flat spot' on the base of the LED as shown here.
- Inserting the LED incorrectly will cause it to not work but will not damage it.
- Push-button light switches like the one seen here were common in the first half of the 20th century (1900-1950's) when they were replaced with toggle switches. One button turned the circuit on while the other turned it off (and popped the other button out). These two projects will make similar switches using the micro:bit buttons.



6. Part 1: The 2-button on-off switch

Start a new Python program (SWITCH1) and add:

```
from microbit import *
```

Remember to get the statement from **<Fns...> Modul <Add-On>**

Tip: This statement tests to see if the micro:bit is present. If the message 'micro:bit not connected' ever appears, just unplug the micro:bit from the calculator and plug it in again (reset).

After the **microbit import** statement is in the Editor, press **[math]** to see the **micro:bit...** menu item. Select it to also add the **buttons** module and then the I/O **pins** module to your program.

```
EDITOR: SWITCH1  
PROGRAM LINE 0004  
from microbit import *  
from mb_buttons import *  
from mb_pins import *  
-  
Fns... a A # Tools Run Files
```

10 Minutes of Code: Python Modules

TI-84 PLUS CE PYTHON

7. Make a *while not escape* loop using the command found on
[math] > micro:bit... > Commands

while not escape():



We also added a comment and a **print()** statement to explain the program.

Note that **while not escape():** is included in the **micro:bit Commands** sub-menu. There's no need to import the **ti_system** module.

8. In the loop body, add two **if** statements, one for each micro:bit button using the two **_was_pressed()** functions:

◆ ◆ **if button_a.was_pressed():**



◆ ◆ **if button_b.was_pressed():**



if . . is found on <Fns...> Ctl.

The two button functions are found on

[math] buttons and touch...

under the **Button_a** and **Button_b** sub-menus.

One button will turn the LED on and the other will turn it off.

9. The LED lights up when it gets power from the micro:bit. We will use digital values for turning it on and off.

To turn the LED on using button A, use the function:

pin0.write_digital(1)

(if your LED is connected to the pin0 port of the expansion board) in the first **if** block.

pin0 is a special micro:bit variable found on

[math] I/O pins under the **Pins** sub-menu

.write_digital() is on the [math] I/O pins Digital menu.

Add the argument **1** to the function.

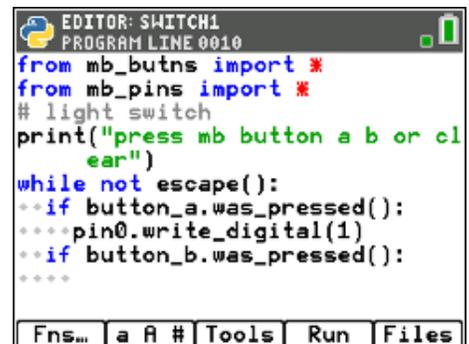
Can you figure out how to turn the LED off? See the next step...



```
EDITOR: SWITCH1
PROGRAM LINE 0007
from microbit import *
from mb_butns import *
from mb_pins import *
# light switch
print("press mb button a b or cl
ear")
while not escape():
..
```



```
EDITOR: SWITCH1
PROGRAM LINE 0011
from mb_butns import *
from mb_pins import *
# light switch
print("press mb button a b or cl
ear")
while not escape():
--if button_a.was_pressed():
....
--if button_b.was_pressed():
....
```



```
EDITOR: SWITCH1
PROGRAM LINE 0010
from mb_butns import *
from mb_pins import *
# light switch
print("press mb button a b or cl
ear")
while not escape():
--if button_a.was_pressed():
....pin0.write_digital(1)
--if button_b.was_pressed():
....
```

10 Minutes of Code: Python Modules

TI-84 PLUS CE PYTHON

10. In a similar manner, turn the LED off when button B was pressed. Set the digital value to **0** as shown here.

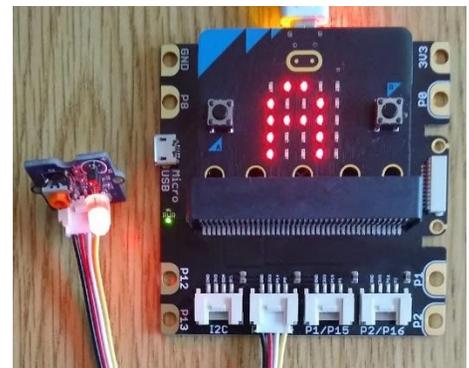
Test your program now to be sure the LED is working properly.

MICRO:BIT: THE LIGHT SWITCH

```
EDITOR: SWITCH1
PROGRAM LINE 0010
from mb_butns import *
from mb_pins import *
# light switch
print("press mb button a b or c l
ear")
while not escape():
  if button_a.was_pressed():
    pin0.write_digital(1)
  if button_b.was_pressed():
    pin0.write_digital(0)
```

11. Make one enhancement: display the letter of each button on the micro:bit display: Add **display.show("A")** when button A is pressed and a similar statement for button B. Remember that you will have to import the micro:bit **display** sub-module to your program to use the **display.show()** function.

After the **while** loop ends (by pressing **[clear]**), make sure that the LED is off and that the display is cleared (or display the special TI logo image using **display.show(ti)** to return the micro:bit display to its original state).



12. Part 2: The ON button

This program will light the LED when button A is *held down* and it will be off when the button is released. Make a copy of your program from Part 1 of this activity. Use **[Files] > Manage (point to switch1) and select Replicate Program**

Give the new program a name (ours is **SWITCH2**)

13. In the new program, modify the button A **if** statement: change **was** to **is**.

if button_a.is_pressed():

```
EDITOR: SWITCH2
PROGRAM LINE 0007
from microbit import *
from mb_butns import *
from mb_pins import *
# light switch
print("press mb button a b or c l
ear")
while not escape():
  if button_a.was_pressed():
    pin0.write_digital(1)
  if button_b.was_pressed():
    pin0.write_digital(0)

EDITOR: SWITCH2
PROGRAM LINE 0009
from microbit import *
from mb_butns import *
from mb_pins import *
# light switch
print("press mb button a b or c l
ear")
while not escape():
  if button_a.is_pressed():
    pin0.write_digital(1)
  if button_b.was_pressed():
    pin0.write_digital(0)
```

10 Minutes of Code: Python Modules

TI-84 PLUS CE PYTHON

14. We will not need the button B condition, but we will need to turn the LED off when button A is *not* pressed, so replace the `if button_b...` statement with an `else:` clause for `if button_a...` and turn off the LED in the `else:` block.

```
if button_a.is_pressed():
    pin0.write_digital(1) # on
else:
    pin0.write_digital(0) # off
```

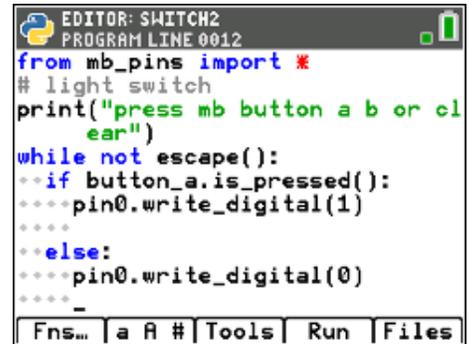
Note that if `button_b...` is not needed and is now commented and `else:` lines up with `if` !

If you displayed the letter “B” in the first project, change it to a space “ ” here so that nothing is displayed when no button is pressed.

15. **Challenge:** Make a single-button on-off switch. Can you modify the program again to turn the LED on when button A is pressed once and turn it off when button A is pressed again? That is, button A toggles the LED on and off by itself and does not need to be held down. Button B is not used at all. If you have micro:bit **version 2**, try using the `pin_logo` (the gold oval above the 5x5 display in the image).

The display can show a ‘1’ when the LED is on and ‘0’ when it is off.

MICRO:BIT: THE LIGHT SWITCH



```
EDITOR: SWITCH2
PROGRAM LINE 0012
from mb_pins import *
# light switch
print("press mb button a b or cl
ear")
while not escape():
    if button_a.is_pressed():
        pin0.write_digital(1)
    else:
        pin0.write_digital(0)
    _
```

