Microbit Lesson 2: Conditional execution

Lesson overview

This lesson:

- The idea of conditional execution
- Syntax: The colon and indentation
- if statement
- while statement
- Microbit feature: Buttons
- Microbit feature: Touch pins

Video

- Microbit lesson 2 video

Conditional execution: If

Recall last lession we said: *statements are executed in order, top down - unless otherwise directed *. We will now look at the "unless otherwise directed" bit.

An `if` statement asks a question. If the answer is `True`, it will run the intended section that immediately follows. Stop intending to return to normal. If the answer is `False`, Python will skip over any new intended section and look for the next line that matches the previous level of indenting.

```
from microbit import *

display.scroll("How awesome is Python?", delay=75)
if button_a.was_pressed():
    display.show(Image.HAPPY)
if button_b.was_pressed():
    display.show(Image.SAD)
```

- What happens if you don't press either button?
- What happens if you press both buttons?

Note: The `delay=75` parameter instructs the Microbit how fast to scroll text. The default speed is 150, any number smaller than that will scroll faster.

We can extend this program as follows...

```
from microbit import *

display.scroll("How awesome is Python?", delay=75)
```

```
if button_a.was_pressed():
    display.show(Image.HAPPY)
    sleep(2000)
    display.scroll("Glad you think Python is amazing", delay=75)
if button_b.was_pressed():
    display.show(Image.SAD)
```

In this instance, we are instructing Python to run three commands if button A was pressed. We can provide as many commands as we wish, provided we match the same level of indentation.

Conditional execution: While

Consider this one change...

```
from microbit import *

while True:
    display.scroll("How awesome is Python?")
    if button_a.was_pressed():
        display.show(Image.HAPPY)
        sleep(2000)
        display.scroll("Glad you think Python is amazing", delay=75)
    if button_b.was_pressed():
        display.show(Image.SAD)
```

A `while` statement works effectively the same as the `if` statement, in that it poses a question and if the response is `True` it will execute the indented code, and if the answer is `False` it will skip the intended code. The key difference is that if it *did* run the indented code, prior to moving on, Python will cycle back to the `while` statement and re-ask the question. This has the effect of creating a looping effect so the indented code will repeatedly run until the question responses with `False`.

One side-effect of the above `while` loop is that because we have specified the condition response as `True`, it will never quit the loop. We could modify it....

```
from microbit import *

while not pin0.is_touched():
    display.scroll("How awesome is Python?")
    if button_a.was_pressed():
        display.show(Image.HAPPY)
        sleep(2000)
        display.scroll("Glad you think Python is amazing", delay=75)
    if button_b.was_pressed():
        display.show(Image.SAD)
display.scroll("Bye!")
```

We have now provided a condition with which to exit the loop. Execute this program and touch both the GND pin and the 0 pin on the Microbit at the same time to terminate the loop.

Nested if's

We can put "if" statements inside other "if" statements. Consider how this would behave?

```
from microbit import *

display.scroll("Whose your Marvel character?")
display.scroll("Goodie or badie?")
if button_a.was_pressed():
    display.scroll("Boy or girl?")
    if button_a.was_pressed():
        display.scroll("Iron man")
    elif button_b.was_pressed():
        display.scroll("Captain Marvel")
elif button_b.was_pressed():
    display.scroll("Secretly good?")
    if button_a.was_pressed():
        display.scroll("Nebula")
    elif button_b.was_pressed():
        display.scroll("Thanos")
```

Activity: 20 questions

By creating a series of yes/no questions using if statements inside of if statements, create a reduced version of a 20 questions game.