Demonstrating Zero G in Free Fall

An astronaut on the space station feels weightless. They still have weight but because the spacecraft they are in is falling towards the Earth at the same velocity as them they cannot feel the weight. Whereas you sitting on your chair can feel the weight of your body pulling you down (but at the same time the Earth is pushing you up). If you were to jump out of an aircraft you too could feel weightless before the parachute opens.

The weight you feel is due to the Earth's attraction to you. And your attraction to Earth. This you feel as gravity which is given a value of 1g. When freely falling there is no gravity as you are in free fall towards the Earth and it does not return until you are once again on the surface (in one piece) and the Earth is pushing up on you.

Zero G can be demonstrated with any device that can be dropped. Mobile phones have accelerometers in them and can be made to give you the values as they are falling. But they are expensive to drop and it is also harder to get the information when they are falling. But the Microbit also has an accelerometer built in and it also has the ability to radio readings to another whilst in free fall.

Equipment Needed

2 Microbits
Battery Pack
Programming Lead
Bubble wrap (or something soft to wrap the Microbit up in)
Elastic Band
Mu Editor

Test the Accelerometer

On the computer open the Mu editor.

One of the Microbits will become the free falling gravity sensor. Connect this to the programming lead and computer.

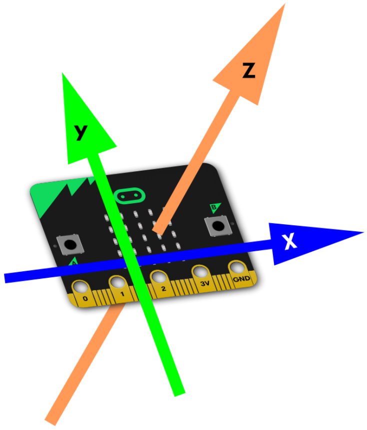
In a clear tab on Mu type the following code:

```
from microbit import *

while True:
    x, y, z = accelerometer.get_values()
    print((x, y, z))
    sleep(250)
```

The code uses a while True forever loop. This constantly runs again and again with a 250 millisecond pause. There are three values to get from the accelerometer which are shown in the image below. These are assigned to x, y and z variables.

Click on the Flash button to copy the code onto the Microbit. Once copied successfully click on Plotter button and then press the Reset button on the back of the Microbit. The values from the accelerometer should now display on the Mu as a series of three lines on the graph.



A value of +1024 or -1024 are the maximum and minimum values depending on how the Microbit is positioned. Move it around and see how the values change.

Reading Gravity

The read out above gives the values for each of the three axis on the Microbit, x, y and z. With a bit of pythagorean maths it is combine the values to give a single value for gravity. It will not be the same value as G as the Microbit has not been calibrated but when there is none then zero is the same.

Alter the code you already have so that it now reads as follows:

```
from microbit import *
from math import sqrt

while True:
    x, y, z = accelerometer.get_values()
    gravity = round(sqrt((x**2)+(y**2)+(z**2)))
    print((gravity,))
    sleep(250)
```

The changes to the code utilise some maths. The sqrt is square root. And the values such as x**2 is squaring the value of x. Round rounds the values down without any decimal places. Watch out for the double brackets and comma in the print line. They are for the plotter which requires a group of figures (a tuple) separated by commas. One figure and a comma is enough though.

As before look at the plot. Now there is just a single combined line. Move the Microbit around again to see the result.

Sending the Results by Radio

So there is a working sensor but it is attached to the computer. Now you need to be able to detach it so that you can make it free fall.

Again **adjust** the code to read as below:

```
from microbit import *
from math import sqrt
import radio

radio.config(channel=7)
radio.on()

while True:
    x, y, z = accelerometer.get_values()
    gravity = round(sqrt((x**2)+(y**2)+(z**2)))
    radio.send(str(gravity))
    sleep(5)
```

This new code brings in the radio. If you are the only person doing this exercise then the radio.config line is not needed. If there are others then between you decide on the channel number you will be using that will be different to theirs. Numbers from 1 to 99 are allowed. Also the radio can only send text not numbers so the value that is the reading is changed into a string of numbers with str. The receiver in a moment will then change it back into the values. The sleep has also been shortened so that more readings are sent per second.

Flash the code swap the Microbits.

Receiving the Values

Open a new tab on the Mu editor.

The code is similar to the previous code but now there is a receive for the radio instead of send.

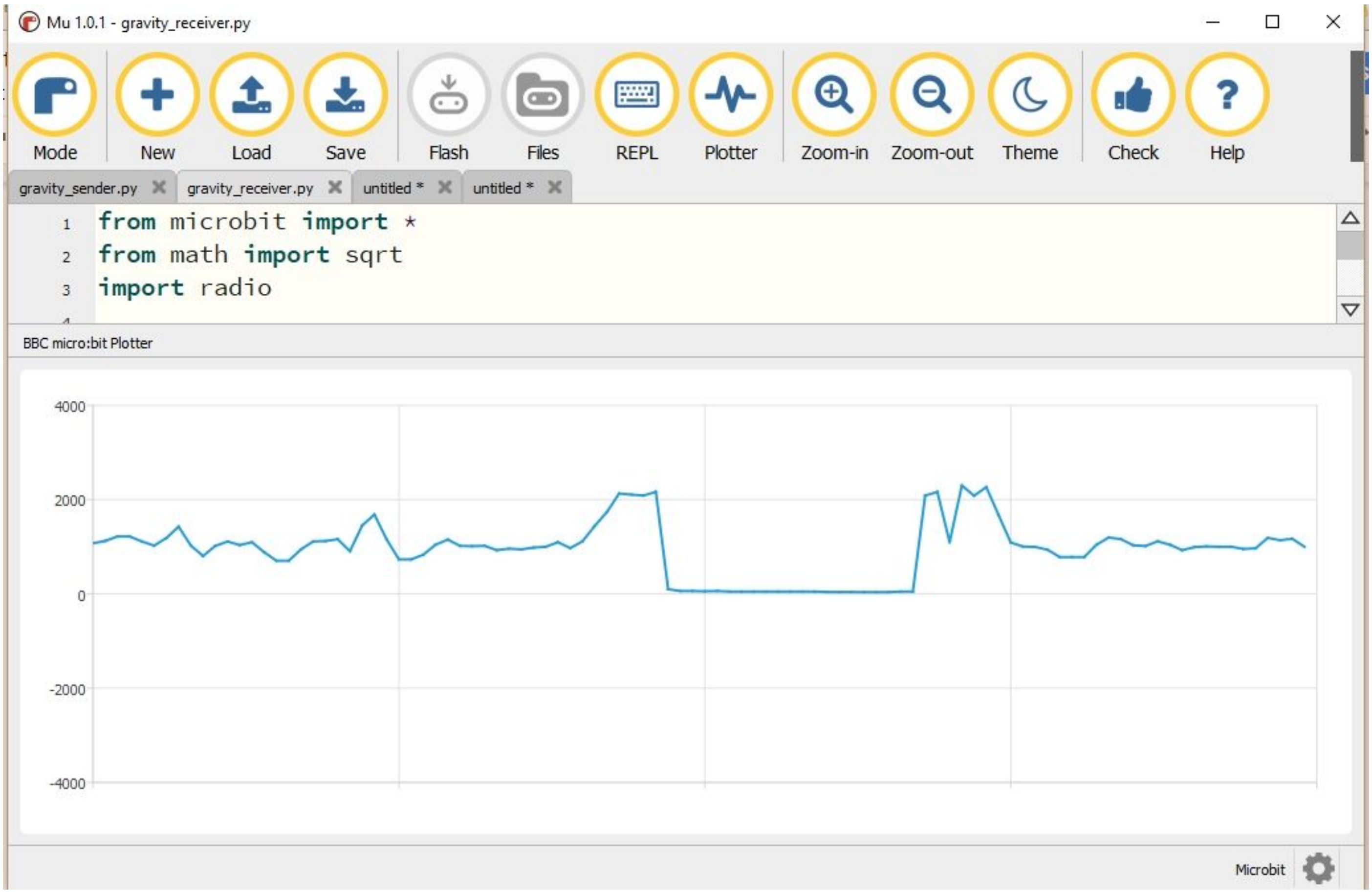
```
from microbit import *
import radio

radio.config(channel=7)
radio.on()

while True:
    message = radio.receive()
    if message:
        message = int(message)
        print((message,))
```

This time the code constantly listens for a message sent from another microbit on the same channel. If there is a message then the message is changed from a string of figures to an integer of numbers. Then the print line sends it to the plotter.

Flash the code to the Microbit and start the plotter as before. Attach the battery pack to the other Microbit, wrap it in bubble wrap and secure with the elastic band. The accelerometer values should now be being sent to the computer by radio. Move it around and see how the values change. If you gently throw the Microbit up and catch it you will see the value on the graph drop to zero.



```
from microbit import *
from math import sqrt
import radio
```