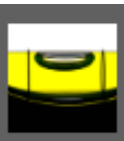
Mission 11:
Spirit Level

Student Workbook

FIRIA LABS

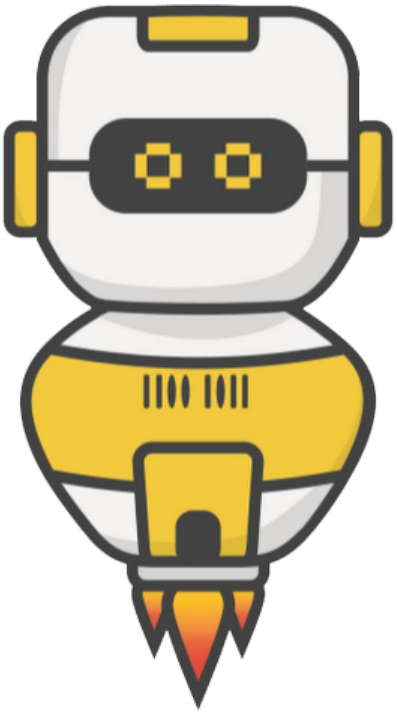


Mission 11: Spirit Level

Create a digital level using the CodeX's built-in accelerometer.

Let's level up!

In this project you'll build a spirit level! This is more than just a fun project – it's a useful tool with practical applications.



Go to the Mission 11 Log and fill out the Pre-Mission preparation.

- Your cell phone can detect if it is level or tilted. What other devices use some kind of sensor to determine their orientation?

Mission 11: Spirit Level

How level is your desk or table? Write some code to find out. During this project you will build a spirit level.

Project Goals:

- Display a numeric "tilt" value from the accelerometer.
- *Scale* the raw tilt value to show 0° to 90° incline.
- Replace the number display with a **graphical** *ball* simulation!

Mission 11: Get started

- Go to https://make.firialabs.com/ and log in.
- Go to Mission 11 
- Click **NEXT** and start Mission 11.

Objective #1: Accel

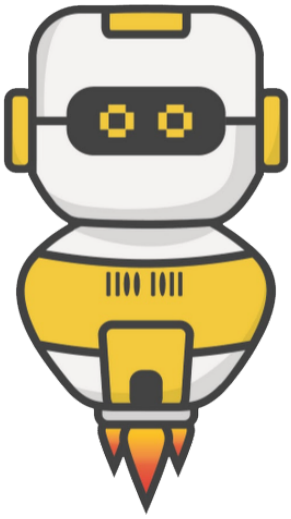
This mission will use CodeX's built-in accelerometer

- What is an accelerometer?
- Click on 

- Go to the Mission Log and answer the question.

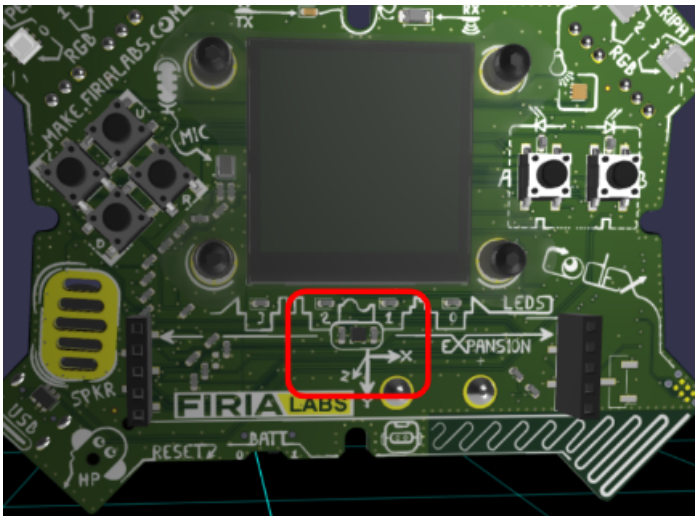
- Return to the digital instructions.
- Go back to the Mission Log and answer the question.

Objective #1: Accel



DO THIS:

- Close the instruction panel
- Use the camera controls to zoom in
- Click on the accelerometer
- Create a new file named **Spirit_Level**



Objective #2: Tilt-o-Matic

When you read the accelerometer, it returns three values, one for each axis.

- The three values are enclosed in parenthesis: (x, y, z)
- This type of data is called a **tuple**.
- Other tuples:
  - Color values: (red, green, blue)
  - Location on a grid: (x, y)

To read the accelerometer, use this code:

```
val = accel.read()
```

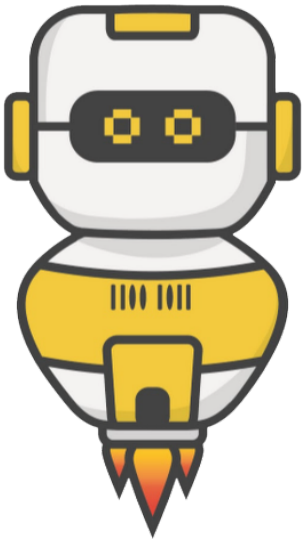
A tuple is a special kind of list. You can get specific x, y or z values like items in a list:

```
tilt_x = val[0]  # first value is at index 0
tilt_y = val[1]
tilt_z = val[2]
```

Objective #2: Tilt-o-Matic

This mission will use CodeX's built-in accelerometer

- Go to the Mission Log and answer the questions about data from the accelerometer.



DO THIS:

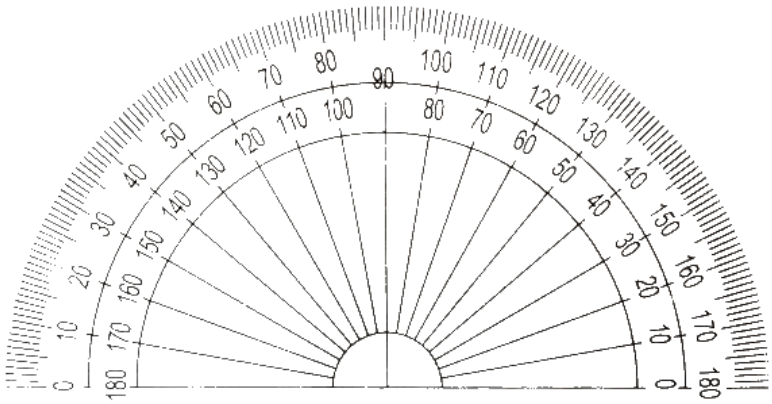
- Read the accelerometer
- Display the x value
- Delay a short time to read the display
- Your code could look like this:

Spirit_Level

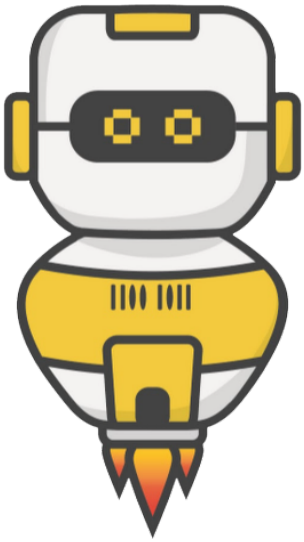
```
1  from codex import *
2  from time import sleep
3
4  while True:
5      val = accel.read()
6      tilt_x = val[0]
7      display.print(tilt_x)
8      sleep(0.5)
```

Objective #3: Scale to Degrees

- The numbers you see on the display are all over the place.
- We usually measure angles in degrees.
- You need to convert the raw data -- numbers you see on the display -- into degrees.
- This involves some complicated math -- trigonometry -- so use the code given and don't worry about the actual math.



Objective #3: Scale to Degrees



DO THIS:

- Import the math module
- Add the code as shown
- Change the display.print to show degrees
- Run the code
  - Can you get CodeX level, with 0 degrees?
- Go to the Mission Log and answer the question

```
from codex import *
from time import sleep
import math

while True:
    val = accel.read()
    tilt_x = val[0]

    scaled = (tilt_x / 16384)
    scaled = min(max(scaled, -1), 1)
    degrees = math.asin(scaled) * 180 / math.pi
    degrees = int(degrees)

    display.print(degrees)
    sleep(0.5)
```

Objective #4: Static Ball

Time to learn a little about drawing on the display

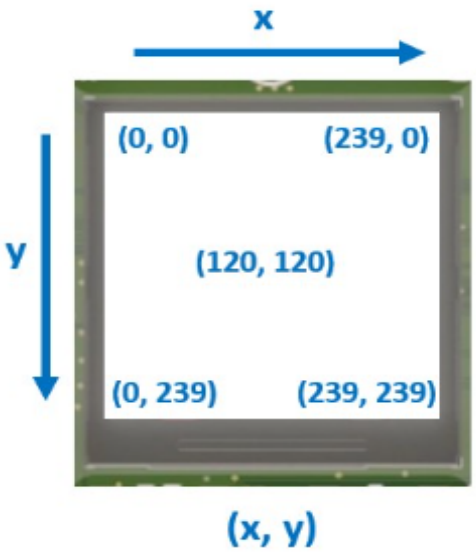
Here are the drawing functions you will use for your spirit level:

Function	Description
`display.fill(color)`	Fill the display with a **color**
`display.draw_line(x1, y1, x2, y2, color)`	Draw a line from (x1, y1) to (x2, y2)
`display.draw_circle(x, y, radius, color)`	Draw a circle with center at (x, y)

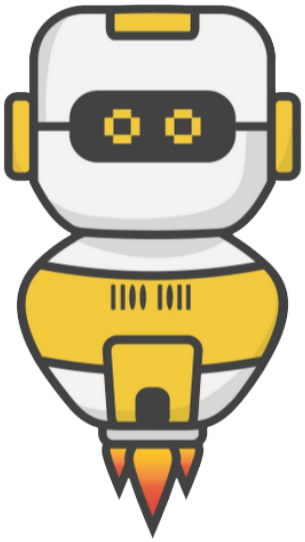
Concept: The display

The CodeX LCD display is 240 pixels x 240 pixels

- Each *tiny* pixel works JUST like the 4 RGB *LED pixels* at the top of the CodeX.
- *x* in the (x, y) is the display *width*
- *y* is the display *height*



Objective #4: Static Ball



DO THIS:

- Define a value for the center of the display
- Type the code to draw a line in the center
- Change the **display.print()** to drawing a circle instead

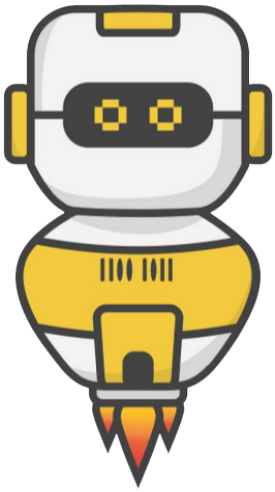
```
from codex import *
from time import sleep
import math

CENTER = 120

display.fill(WHITE)
display.draw_line(CENTER, 0, CENTER, 105, BLACK)
display.draw_line(CENTER, 135, CENTER, 239, BLACK)

while True:
    val = accel.read()
    tilt_x = val[0]

    scaled = (tilt_x / 16384)
    scaled = min(max(scaled, -1), 1)
    degrees = math.asin(scaled) * 180 / math.pi
    degrees = int(degrees)

    display.draw_circle(CENTER, CENTER, 15, ORANGE)
    sleep(0.5)
```

Objective #5: Rolling Stone

Time to make that ball move

- Use the information you have:
  - Center of display
  - Degrees for x
- Use math to calculate the position of the circle
- Use a variable for the data

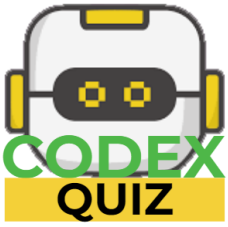
Objective #5: Let's Keep Playing



DO THIS:

- Define a value for the x position
- Assign it the value:
  - x = CENTER + degrees
- Use the variable in the draw_circle() command
- Run the code and tilt CodeX to see the ball move

```
while True:
    val = accel.read()
    tilt_x = val[0]

    scaled = (tilt_x / 16384)
    scaled = min(max(scaled, -1), 1)
    degrees = math.asin(scaled) * 180 / math.pi
    degrees = int(degrees)

    x = CENTER + degrees
    display.draw_circle(x, CENTER, 15, ORANGE)
    sleep(0.5)
```



Mission Quiz: Accelisplay

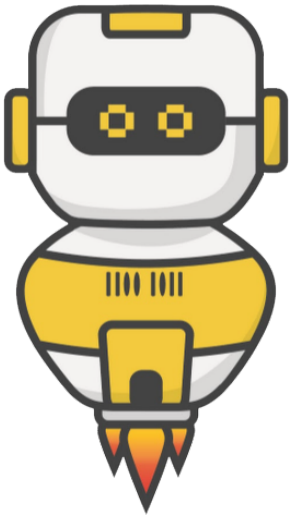
Test your skills by **taking the quiz**.

Objective #6: Eraser First

Spirit Level -- final touches

- Why is the ball drawing on top of itself?
- Because you are not erasing it first.
- Cover your tracks!
- Draw a white circle to “erase” the current circle before moving to a new position and drawing a new orange circle.

Objective #6: Reduce Repetition



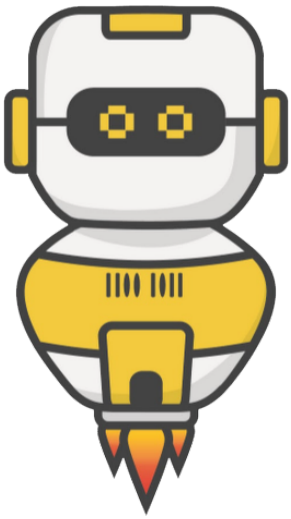
DO THIS:

- Define the x variable above the while loop
- Draw a white circle before calculating a new x value
- Run your program
  Does it work correctly?

```
CENTER = 120

display.fill(WHITE)
display.draw_line(CENTER, 0, CENTER, 105, BLACK)
display.draw_line(CENTER, 135, CENTER, 239, BLACK)

x = CENTER

while True:
    val = accel.read()
    tilt_x = val[0]

    scaled = (tilt_x / 16384)
    scaled = min(max(scaled, -1), 1)
    degrees = math.asin(scaled) * 180 / math.pi
    degrees = int(degrees)

    display.draw_circle(x, CENTER, 15, WHITE)
    x = CENTER + degrees
    display.draw_circle(x, CENTER, 15, ORANGE)
    sleep(0.5)
```

Mission Complete

You have completed the eleventh mission.



Do this:

- Read your “Completed Mission” message
- Complete your Mission 11 Log
    - Post-Mission Reflection
- Get ready for your next mission!

Wait! Before you go ... Clear the CodeX

Go to FILE -- BROWSE FILES

Select the “**Clear**” file and open it

Run the program to clear the CodeX

Okay. Now you can go.