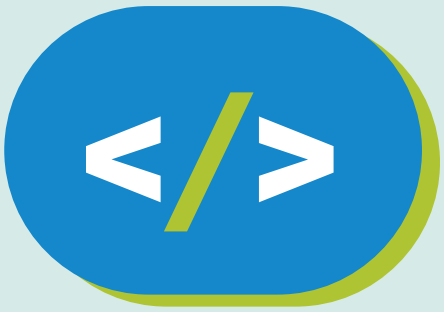


Código Pi
Kit de experimentación

Comenzando a programar la placa Sense HAT

APRENDER CONECTADOS

Ministerio de Educación, Cultura, Ciencia y Tecnología
Presidencia de la Nación

Autoridades

Presidente de la Nación
Mauricio Macri

Jefe de Gabinete de Ministros
Marcos Peña

Ministro de Educación, Cultura, Ciencia y Tecnología
Alejandro Finocchiaro

Secretario de Gobierno de Cultura
Pablo Avelluto

Secretario de Gobierno de Ciencia, Tecnología e Innovación Productiva
Lino Barañao

Titular de la Unidad de Coordinación General del Ministerio de Educación, Cultura, Ciencia y Tecnología
Manuel Vidal

Secretaria de Innovación y Calidad Educativa
Mercedes Miguel

Directora Nacional de Innovación Educativa
María Florencia Ripani

ISBN en trámite

Este material fue producido por el Ministerio de Educación, Cultura, Ciencia y Tecnología de la Nación en el marco del Plan Aprender Conectados.

Cómo empezar a usar la placa Sense HAT

En este proyecto vamos a explorar el *hardware* de la placa Sense HAT y su biblioteca Python. La placa Sense HAT es una parte fundamental que le permite a la Raspberry Pi percibir el mundo que la rodea.
Aprenderemos cómo controlar la matriz led, recopilar datos de sensor,provenientes del mundo que nos rodea y combinaremos estas ideas en algunos pequeños proyectos.

Al aplicar este recurso con Raspberry Pi y Sense HAT aprenderemos a:

- Comunicar con el Sense HAT usando Python.
- Acceder a las salidas del Sense HAT.
- Programar las entradas del Sense HAT.
- Usar la biblioteca Sense HAT para mostrar mensajes e imágenes.
- Controlar la orientación, recopilar datos de sensor y responder al movimiento.
- Usar variables para almacenar datos del sensor.
- Usar bucles para repetir comportamientos.

Qué necesitaremos

Hardware

- Raspberry Pi con una tarjeta SD y los periféricos habituales.
- Placa Sense HAT.

Software

- Python 3
- Sense Hat para Python 3

La última versión de Raspbian, ya incluye los paquetes del *software* requeridos.

Comenzando con la placa Sense HAT

La placa Sense HAT es un complemento Raspberry Pi. La placa permite realizar mediciones de temperatura, humedad, presión y orientación, y generar información utilizando su matriz led incorporada.

Si no tenemos acceso a una placa Sense HAT, podemos usar el recurso del emulador disponible en nuestro dispositivo Raspberry Pi.

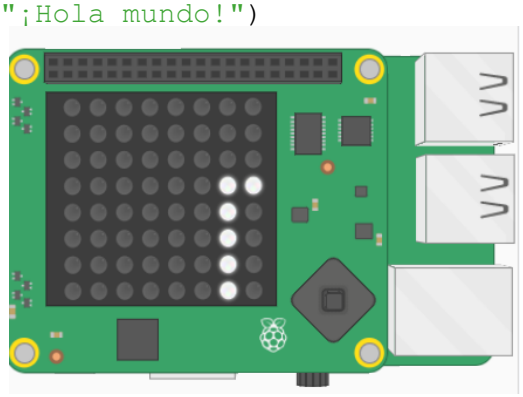
Empecemos abriendo Python 3 en el menú principal, para realizar las siguientes actividades:

Escribiremos un código que desplaza un texto en la matriz led. Este programa contiene dos líneas fundamentales que importan el *software* de Sense HAT y crean un objeto `sense` que representa la Sense HAT.

```
from sense_hat import
SenseHat

sense = SenseHat()
```

Esta tercera línea es la que hace que la placa Sense HAT realice lo siguiente:

```
sense.show_message("¡Hola mundo!")
```



Es muy probable que ya hayamos descubierto que podemos cambiar fácilmente el mensaje con nuestro propio texto, pero hay más cosas que podemos hacer.

1. Podemos expandir el comando `sense.show_message` para que incluya algunos **parámetros** adicionales que cambiarán el comportamiento del mensaje.

Parámetro	Efecto
scroll_speed	El parámetro `scroll_speed` modifica la rapidez con la que se mueve el texto en la pantalla. El valor predeterminado es 0.1. Cuanto más elevado sea el número, más **lenta** será la velocidad.
text_colour	El parámetro `text_colour` modifica el color del texto y se especifica usando 3 valores para Rojo, Verde y Azul. Cada valor puede ser entre 0 y 255, por lo tanto [255,0,255] sería Rojo + Azul = Púrpura
back_colour	El parámetro `back_colour` modifica el color de fondo y funciona de la misma manera que el parámetro `text_colour`.

Así que el siguiente código mostrará el texto `¡Astro PI es genial!` más lento, con el texto en amarillo **[255,255,0]** y el fondo en azul **[0,0,255]**:

```
from sense_hat import SenseHat
sense = SenseHat ()
sense.show_message ("¡Hola mundo!",scroll_speed=0.05,
text_colour=[255,255,0])
```

También podemos hacer que el mensaje se repita usando un bucle *while*, de la siguiente forma:

```
from sense_hat import SenseHat
sense = SenseHat()
while True:
 sense.show_message ("¡Hola mundo,
  scroll_speed=0.05,text_colour=[255,255,0])
```

2. Hagamos clic en **File** ('Archivo') **> Save As (**'Guardar como'**)**, le damos un nombre a nuestro programa; por ejemplo *loop_text.py*, después presionamos **F5** para ejecutar.

3. La matriz led también puede mostrar un solo carácter, en lugar de un mensaje entero, usando la función `sense.show_letter` que también tiene algunos **parámetros** opcionales.

Parámetro	Efecto
scroll_speed	El parámetro `scroll_speed` modifica la rapidez con la que se mueve el texto en la pantalla. El valor predeterminado es 0.1. Cuanto más elevado sea el número, más **lenta** será la velocidad.
text_colour	El parámetro `text_colour` modifica el color del texto y se especifica usando 3 valores para Rojo, Verde y Azul. Cada valor puede ser entre 0 y 255, por lo tanto [255,0,255] sería Rojo + Azul = Púrpura.
back_colour	El parámetro `back_colour` modifica el color del fondo y se especifica usando 3 valores para Rojo, Verde y Azul. Cada valor puede ser entre 0 y 255, por lo tanto [255,0,255] sería Rojo + Azul = Púrpura.

Así que este programa mostraría una única *J* en rojo:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.show_letter("J",text_colour=[255, 0, 0])
```

Y en este programa agregamos la función `sleep` ('pausa'**)** para mostrar letras separadas por una breve pausa:

```
from sense_hat import SenseHat
from time import sleep
sense = SenseHat()
sense.show_letter("O",text_colour=[255, 0, 0]) sleep(1)
sense.show_letter("M",text_colour=[0, 0, 255]) sleep(1)
sense.show_letter("G",text_colour=[0, 255, 0]) sleep(1)
sense.show_letter("!",text_colour=[0, 0, 0], back_colour=[255, 255, 255])
sleep(1)
sense.clear ()
```

Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'**)**, le damos un nombre a nuestro programa; por ejemplo *omg.py*, después presionamos **F5** para ejecutar.

Para que sea más interesante, podríamos usar un generador de números al azar para elegir un número entre 0 y 255 para los colores:

```
from sense_hat import SenseHat from time import sleep
from random import randint

sense = SenseHat()

r=randint(0,255)
sense.show_letter("O",text_colour=[r, 0, 0])

sleep(1)

r=randint(0,255)
sense.show_letter("M", text_colour=[0, 0, r])

sleep(1)

r=randint(0,255)
sense.show_letter("G", text_colour=[0, r, 0]) sleep(1)

sense.show_lettertext_colour=[0, 0, 0], back_colour=[255, 255, 255])
sleep(1)
sense.clear ()
```

4. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'**)**, damos un nombre a nuestro programa; por ejemplo, *random_omg.py*, después presionamos **F5** para ejecutar.

En estos dos programas se usó el método `sense.clear()` al final para borrar la matriz.

Ideas

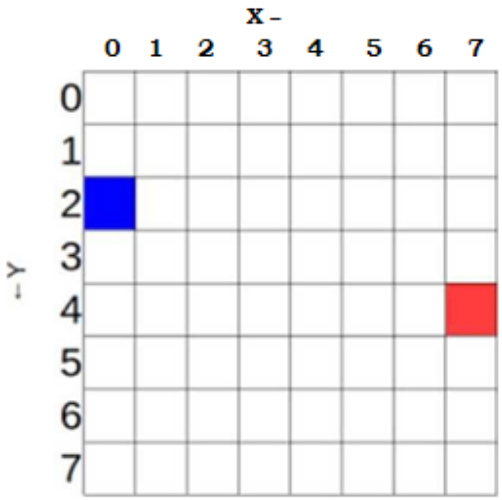
■ ¿Podríamos usar las ideas que aplicamos hasta ahora, para contar un chiste en la pantalla led?
■ Todos los ejemplos que se mostraron hasta aquí, podrían ser más cortos aunque se obtiene el mismo resultado. ¿Podríamos encontrar otras formas para que sean más cortos y más eficaces?
■ ¿Cómo podríamos elegir un color totalmente al azar, en lugar de elegir solo la sombra de un color aleatoriamente?
■ Si nuestra placa Sense HAT está conectada a Internet, ¡podríamos usar una biblioteca de Twitter para que muestre los tweets entrantes!

Visualizar imágenes

La matriz led puede mostrar más que un simple texto: se puede controlar cada led individualmente para crear una imagen. Podemos lograr esto de varias formas.

1. La primera posibilidad es configurar píxeles (led) individualmente; podemos hacer eso usando el método `sense.set_pixel()`. Primero, tenemos que saber claramente cómo describiremos cada píxel.

La placa Sense HAT usa un sistema de coordenadas como el que se muestra a continuación; indefectiblemente la numeración comienza en 0, no en 1. También, el origen es en el extremo **superior izquierdo,** y no en el inferior izquierdo, como quizás estemos acostumbrados.

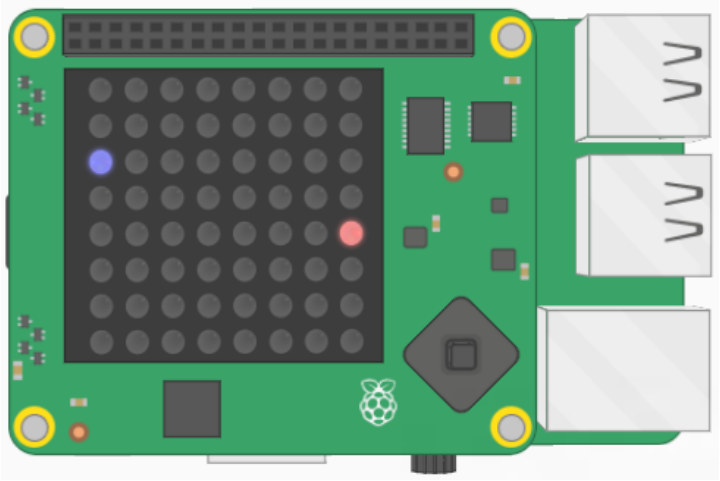


■El píxel azul está en las coordenadas (0,2)

■ El píxel rojo está en las coordenadas (7,4)

Para replicar el diagrama anterior debemos ingresar un programa como el siguiente:

```
from sense_hat import
SenseHat

sense = SenseHat ()

sense.set_pixel (0, 2, [0, 0, 255])
sense.set_pixel (7, 4, [255, 0, 0])
```



¿Podremos adivinar qué crea el siguiente código? Intentemos editarlo.

```
from sense_hat import SenseHat

sense = SenseHat ()

sense.set_pixel (2, 2, [0, 0, 255])
sense.set_pixel (4, 2, [0, 0, 255])
sense.set_pixel (3, 4, [100, 0, 0])
sense.set_pixel (1, 5, [255, 0, 0])

sense.set_pixel (2, 6, [255, 0, 0])
sense.set_pixel (3, 6, [255, 0, 0])
sense.set_pixel (4, 6, [255, 0, 0])
sense.set_pixel (5, 5, [255, 0, 0])
```

2. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *simple_image.py*, después presionamos **F5** para ejecutar.

3. Configurar píxeles individualmente puede funcionar muy bien, pero se complica bastante cuando queremos configurar múltiples pixeles. Hay otro método que puede configurar todos los píxeles de una vez, llamado `sense.set_pixels`. Su uso es bastante fácil; simplemente damos una lista de valores de color para cada píxel en la matriz.

Podríamos ingresar algo parecido a esto...

```
sense.set_pixels([[255, 0, 0], [255, 0, 0], ..., [255, 0, 0])
```

... pero llevaría mucho tiempo y sería muy complejo.

En cambio, podemos usar algunas variables para definir nuestra paleta de colores. En este ejemplo, estamos usando los siete colores del arcoíris (ver referencia):

```
r = [255, 0, 0]
o = [255, 127, 0]
y = [255, 255, 0]
g = [0, 255, 0]
b = [0, 0, 255]
i = [75, 0, 130]
v = [159, 0, 255]
e = [0, 0, 0]
```

Referencia de los colores

- r inicial de *red* que significa 'rojo' en inglés
- o inicial de *orange* que es 'naranja' en inglés
- y inicial de *yellow* que es 'amarillo' en inglés
- g inicial de *green* que es 'verde' en inglés
- b inicial de *blue* que es 'azul' en inglés
- i inicial de *indigo* que es 'índigo' en inglés
- v inicial de *violet* que es 'violeta' en inglés
- e inicial de *empty* que es 'vacío' en inglés y equivale al negro

Luego podemos describir nuestra matriz, creando una lista de nombres de colores en 2D:

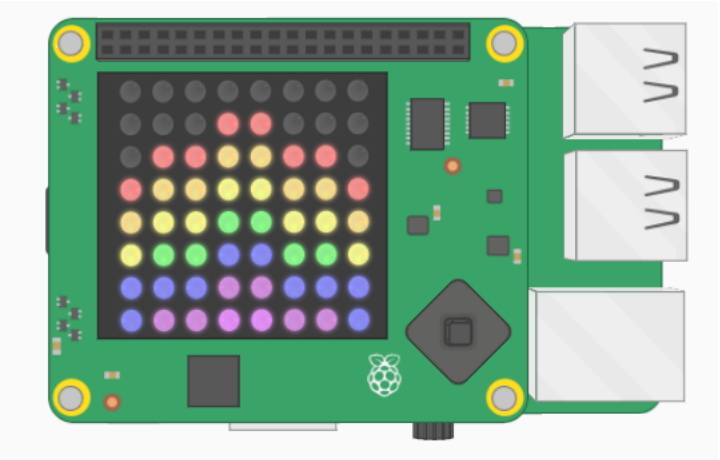
```
Image = [
e,e,e,e,e,e,e,e,
e,e,e,r,r,e,e,e,
e,r,r,o,o,r,r,e,
r,o,o,y,y,o,o,r,
o,y,y,g,g,y,y,o,
y,g,g,b,b,g,g,y,
b,b,b,i,i,b,b,b,
b,i,i,v,v,i,i,b
]
```

Después le damos la lista de `image` (imagen) al método `sense.set_pixels` y dibujamos la imagen. El programa terminado se vería así:

```
from sense_hat import SenseHat
sense = SenseHat()
r = [255,0,0]
o = [255,127,0]
y = [255,255,0]
g = [0,255,0]
b = [0,0,255]
i = [75,0,130]
v = [159,0,255]
e = [0,0,0]

image = [
e,e,e,e,e,e,e,e,
e,e,e,r,r,e,e,e,
e,r,r,o,o,r,r,e,
r,o,o,y,y,o,o,r,
o,y,y,g,g,y,y,o,
y,g,g,b,b,g,g,y,
b,b,b,i,i,b,b,b,
b,i,i,v,v,i,i,b
]
sense.set_pixels(image)
```

4. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa, por ejemplo *rainbow.py*. Luego presionamos **F5** para ejecutar.

Deberíamos ver un arcoíris precioso en nuestra matriz led:



Ideas

• Ahora podemos crear imágenes en nuestra matriz led de dos formas diferentes, intentemos crear nuestras propias imágenes o personajes (*sprites*).

• ¿Podemos alternar entre imágenes para crear una animación?

Configurar la orientación

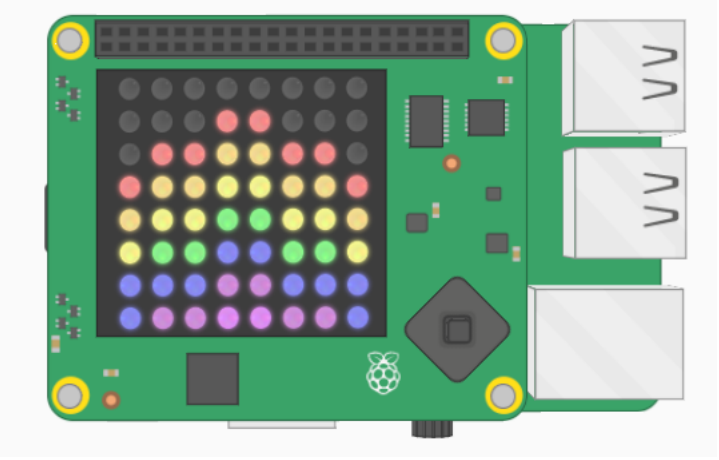
Hasta ahora, todos nuestros textos e imágenes aparecieron de la misma forma, hacia arriba, asumiendo que el puerto HDMI está abajo. Sin embargo, quizás esto no siempre sea así (especialmente en el espacio) así que tal vez querramos cambiar la orientación de la matriz. Para hacer esto, podemos usar el método `sense. set rotation ()` y entre los paréntesis ingresamos uno de los cuatro ángulos (0,90, 180, 270).

Para rotar nuestra pantalla 180º usaremos esta línea:

```
sense.set rotation(180)
```

1. Cuando usemos esta función en el programa de arcoiris, se verá así:

```
from sense_hat import SenseHat

sense = SenseHat()

r = [255,0,0]
o = [255,127,0]
y = [255,255,0]
g = [0,255,0]
b = [0,0,255]
i = [75,0,130]
v = [159,0,255]
e = [0,0,0]


image = [
e,e,e,e,e,e,e,e,
e,e,e,r,r,e,e,e,
e,r,r,o,o,r,r,e,
r,o,o,y,y,o,o,r,
o,y,y,g,g,y,y,o,
y,g,g,b,b,g,g,y,
b,b,b,i,i,b,b,b,
b,i,i,v,v,i,i,b
]
sense.set_pixels(image)
sense.set_rotation(180)
```



2. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *rainbow_flip.py*, después presionamos **F5** para ejecutar.

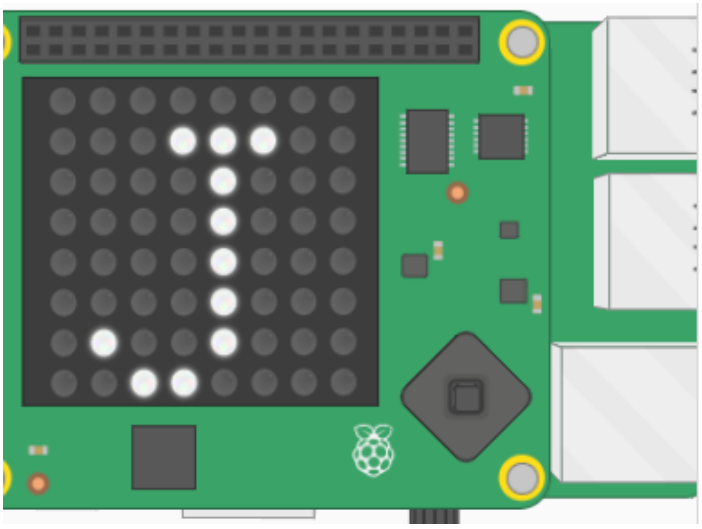
3.También podemos crear texto giratorio usando un bucle `for`:

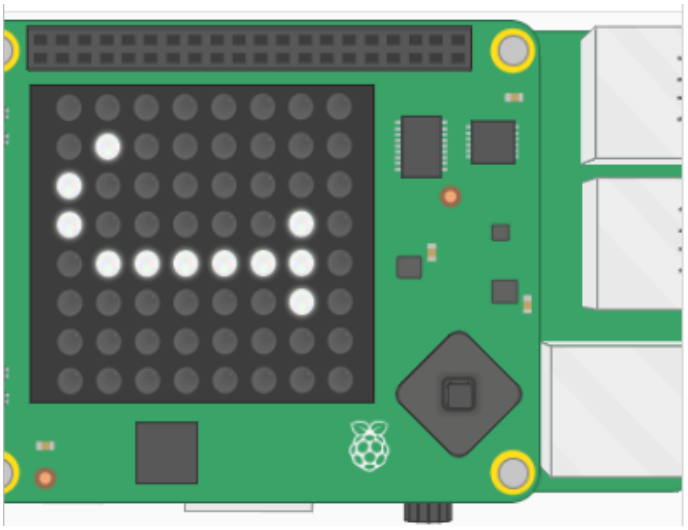
```
from sense_hat import SenseHat
from  time import sleep

sense = SenseHat()

sense.show_letter("J")

angles = [0, 90, 180, 270, 0, 90, 180, 270]
for r in angles:
    sense.set_rotation(r)
    sleep(0.5)
```

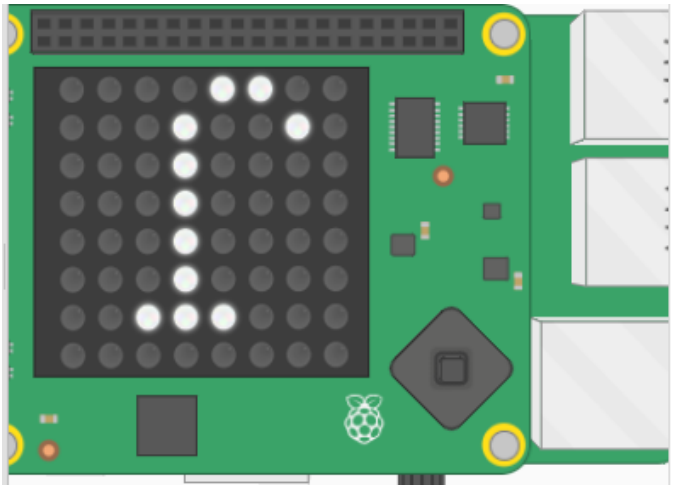
Este programa muestra la letra «J» y después configura la rotación según cada valor que aparece en la lista de ángulos, con una pausa de 0,5 segundos.

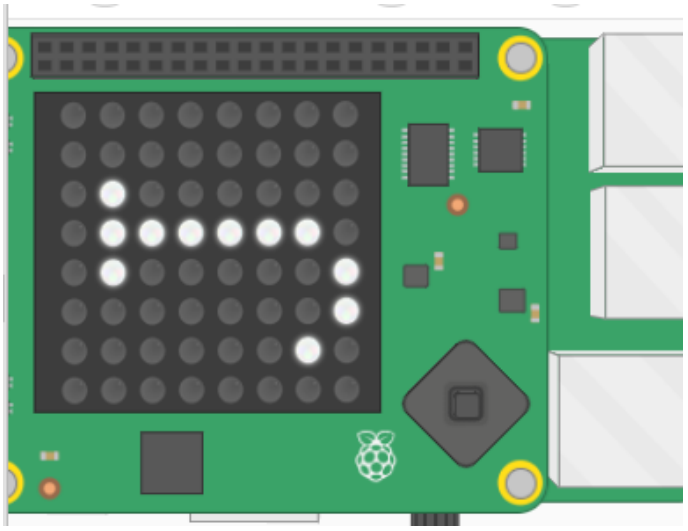
4. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *spinning_i.py*, después presionamos **F5** para ejecutar.

Deberíamos ver dos secuencias de las rotaciones de la J como las siguientes:









5. También podemos girar la imagen en la pantalla, tanto horizontal como verticalmente, usando estas líneas:

```
sense.flip_h()
```

o

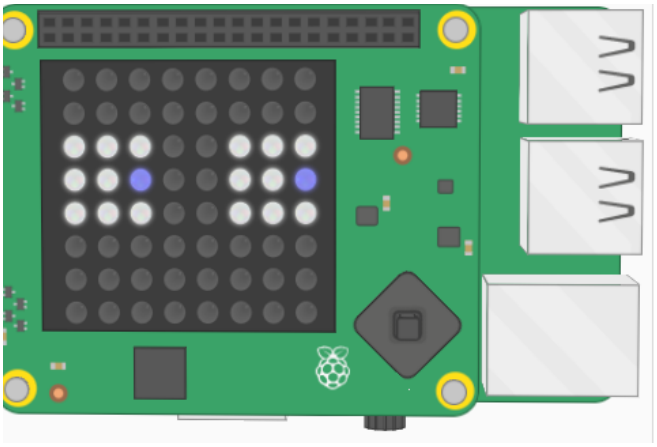
```
sense.flip_v()
```

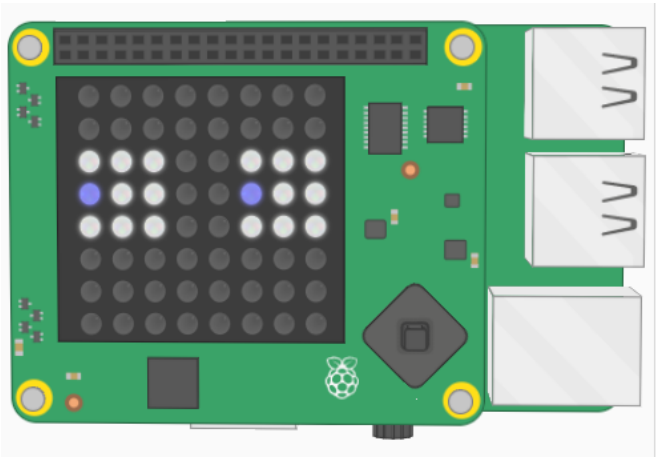
Con el siguiente código podremos crear una sencilla animación invirtiendo horizontalmente la imagen repetidas veces:

```
from sense_hat import SenseHat
from time import sleep

sense = SenseHat()

w = [150, 150, 150]
b = [0, 0, 255]
e = [0, 0, 0]

image = [
e,e,e,e,e,e,e,e,
e,e,e,e,e,e,e,e,
w,w,w,e,e,w,w,w,
w,w,b,e,e,w,w,b,
w,w,w,e,e,w,w,w,
e,e,e,e,e,e,e,e,
e,e,e,e,e,e,e,e,
e,e,e,e,e,e,e,e,
]
sense.set_pixels(image)
while True:
    sleep(1)
    sense.flip_h()
```

6. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *eyes.py*, después presionamos **F5** para ejecutar.

¡Veríamos un continuo movimiento de ojos!

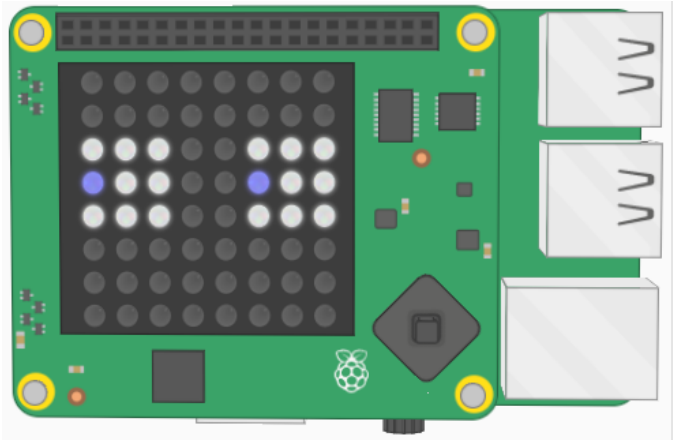




Ideas

■ Podemos crear una imagen giratoria usando una de las técnicas de dibujo que ya mostramos y después usá el método `sense. set_rotation` para hacer que rote.

■ Con todo lo que hicimos hasta ahora, podríamos hacer un dado electrónico (*dice*) como el que mostramos aquí:



Este dado utiliza lo siguiente:

- Visualización de texto.
- Control de tiempo.
- Configuración de la rotación.
- Números aleatorios.
- Variables.

Detectando el medio ambiente

La placa Sense HAT tiene un conjunto de sensores ambientales para detectar las condiciones del entorno. Puede medir:

- presión
- temperatura
- humedad

Podemos obtener estas lecturas usando tres métodos simples:

- `sense.get_temperature()` - Esto mostrará la temperatura en grados Celsius.
- `sense.get_pressure()` - Esto mostrará la presión en milibares.
- `sense.get_humidity()` - Esto mostrará la humedad en porcentaje.

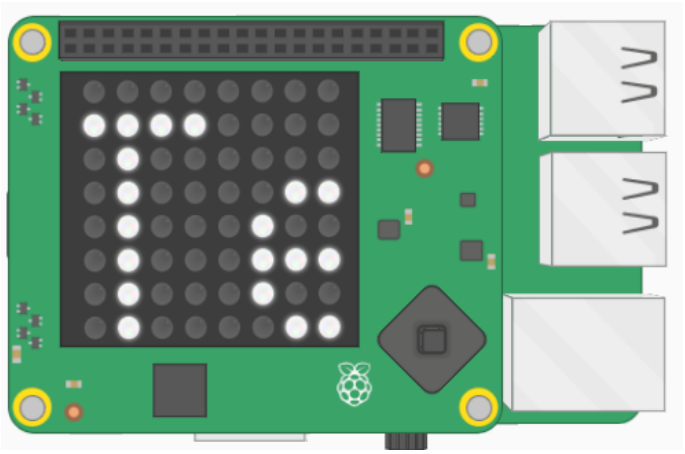
1. Con estos sensores podemos crear mensajes para mostrar en la pantalla con un texto que se vaya desplazando, informando las condiciones de temperatura, humedad y presión actuales.

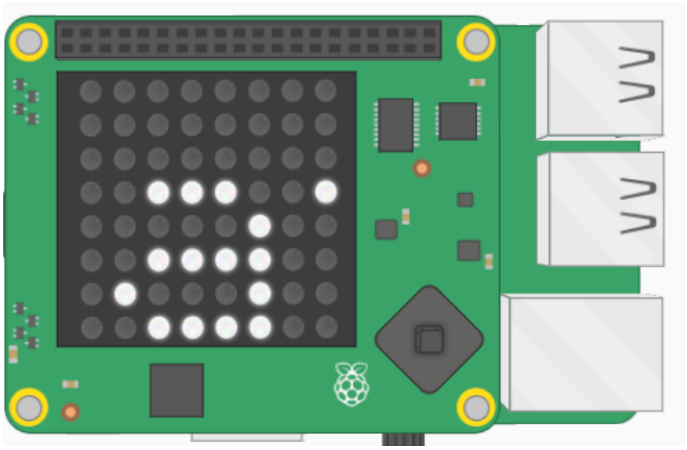
```
from sense_hat import SenseHat
sense = SenseHat()

while True:
    t = sense.get_temperature()
    p = sense.get_pressure()
    h = sense.get_humidity()

# Redondeo de los valores con un decimal
    t = round(t, 1)
    p = round(p, 1)
    h = round(h, 1)

# Creación del mensaje
  msg = "Temperatura = {0}, Presion = {1}, Humedad =
{2}".format(t,p,h)
```

```
# Muestra el mensaje
   sense.show_message(msg, scroll_speed=0.05)
```

2. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *env.py*, después presionamos **F5** para ejecutar.

Deberíamos ver dos secuencias de las rotaciones de la J como las siguientes:





3. Podríamos usar algún color para saber si las condiciones ambientales están dentro de los rangos normales.

Según los siguientes parámetros:

- Temperatura (18,3 a 26,7 grados Celsius)
- Presión (979 a 1027 milibares)
- Humedad (alrededor del 60%)

Podríamos usar una instrucción `if` para verificar estas condiciones y configurar un color de fondo para el desplazamiento:

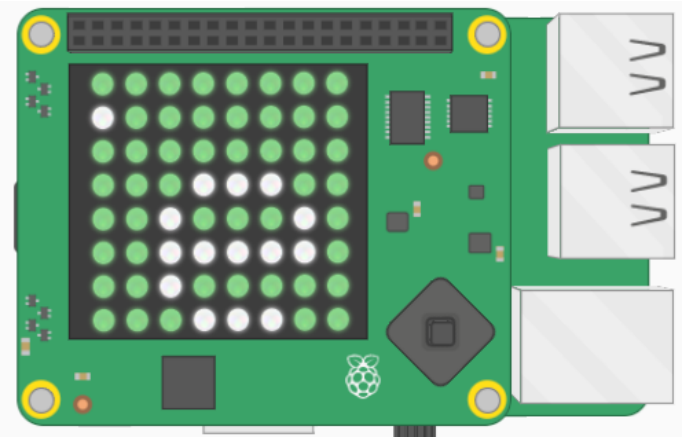
```
if t > 18.3 and t < 26.7:
    bg = [0, 100, 0] # se asigna verde al fondo de pantalla
    else:
    bg = [100, 0, 0] # se asigna rojo al fondo de pantalla
```

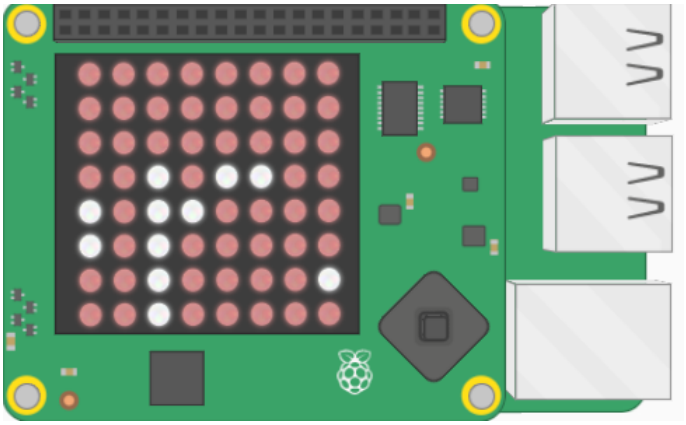
Nuestro programa completo se vería así:

```
from sense_hat import SenseHat
sense = SenseHat()
while True:
      t = sense.get_temperature()
      p = sense.get_pressure()
      h = sense.get_humidity()
      t = round(t, 1)
      p = round(p, 1)
      h = round(h, 1)
      if t > 18.3 and t < 26.7:
     bg = [0, 100, 0]   # verde
else:
      bg = [100, 0, 0]    # rojo
msg = "Temperatura = [0}, Presion = {1}, Humedad = {2}". format (t, p, h)
sense.show_message(msg, scroll_speed=0.05, back_colour=bg)
```

4. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *scrolling_env.py*, después presionamos **F5** para ejecutar.

Dependiendo del valor de la temperatura, veríamos el mensaje con fondo verde o rojo.





Ideas

■ Actualmente el programa con el mensaje advierte poniendo la pantalla en rojo, cuando se presenta una temperatura anormal. ¿Podemos agregar el mismo comportamiento para la presión y para la humedad?

■ Podríamos crear un termómetro gráfico simple que arroje diferentes colores o patrones según la temperatura.

■ Si todavía no lo hiciste, probá con una botella y el sensor de presión.

Detectar movimiento

La placa Sense HAT tiene un conjunto de sensores que pueden detectar movimiento. Tiene un chip IMU (unidad de medición inercial).

En realidad, son tres sensores en uno:

● Un giroscopio (para detectar en qué dirección está orientada la placa) que mide el momento y la rotación.

● Un acelerómetro (para detectar movimiento) que mide las fuerzas de aceleración; se puede usar para encontrar la dirección de la gravedad.

● Un magnetómetro (para detectar campos magnéticos), que mide el propio campo magnético de la Tierra, de manera similar a la de una brújula.

¿Por qué es importante un sensor de movimiento? En ciertas ocasiones, hay una cuestión de importancia absoluta para la que siempre hay que conocer la respuesta: ¿cómo estoy orientado?

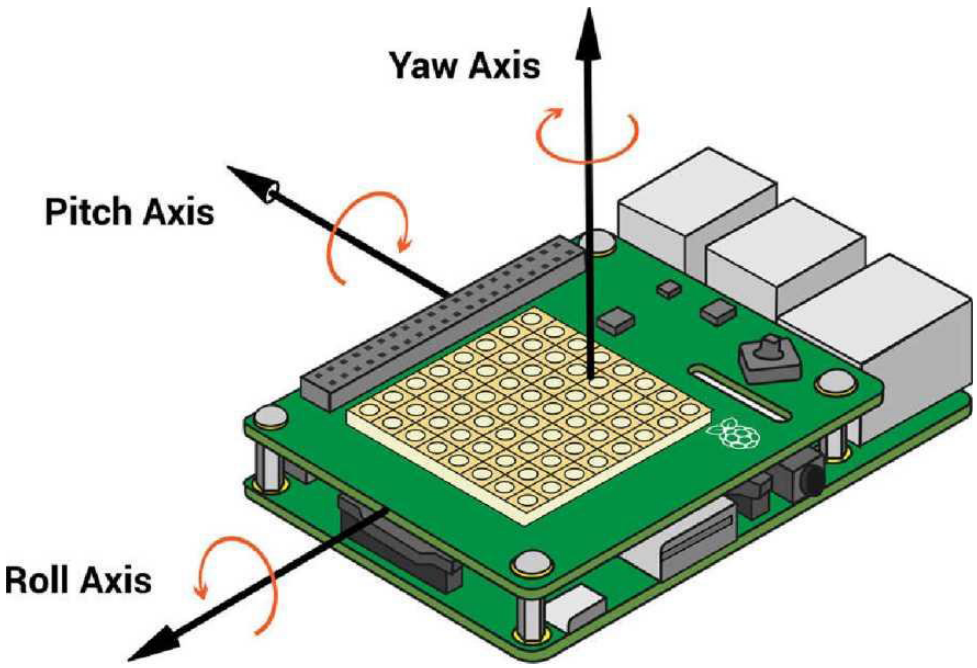
¡No conocer la orientación es un gran problema! Por eso el sensor IMU como el del Sense HAT se usa, por ejemplo, en todas las naves tripuladas y no tripuladas para rastrear movimientos y mantener una comprensión de la orientación. Incluso las primeras naves espaciales las tenían. Un ejemplo de ello fue el programa Apolo, que permitió a los humanos aterrizar en la superficie de la Luna varias veces.

La de arriba es una imagen del sensor IMU del módulo de comando Apolo. Notarás lo grande que es, comparado con el que se encuentra en la placa SenseHat: esa es la diferencia entre la tecnología de 1975 y de la actualidad.

Antes que continuemos con la experimentación de la detección de movimiento, es importante comprender tres términos claves:

Los tres ejes que se usan para describir el movimiento son:

- **Lateral (Pitch)** (como el movimiento de un avión cuando despega).
- **Longitudinal (Roll)**, (cuando el avión hace un giro de 360º sobre sí mismo).
- **Vertical (Yaw), (**imaginen manejando el avión como un auto).



Podemos averiguar la orientación de la placa Sense HAT usando el método `sense.get_orientation()`:

```
orientation =
sense.get_orientation()
pitch = orientation['pitch']
roll = orientation['roll']
yaw = orientation['yaw']
```

Esto obtendrá los tres valores de orientación medidos en grados y los guardará en las tres variables `pitch`, `roll` y `yaw`.

1.Podemos explorar estos valores a través de este simple código:

```
from sense_hat import SenseHat
sense = SenseHat()
while True:
    orientation = sense.get_orientation()
    pitch = orientation['pitch']
    roll = orientation['roll']
    yaw = orientation ['yaw']
   print("pitch={0}, roll={1}, yaw={2}".format(pitch,yaw,roll))
```

1. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *orientation.py*, después presionamos **F5** para ejecutar.

Nota: Cuando usemos los sensores de movimiento, es importante realizar lecturas frecuentes de los datos, utilizando un ciclo como while true. Si realizamos las lecturas muy espaciadas (por ejemplo, agregando un **time. sleep (0.5)** en el ciclo), obtendremos resultados extraños. Esto es porque el código necesita muchas mediciones para combinar correctamente los datos que arrojan el giroscopio, el acelerómetro y el magnetómetro.

2. Otra forma de detectar la orientación es usar el método `sense. get_accelerometer_raw ()` que nos dice la cantidad de fuerza G que actúa en cada eje (*x*, *y*, *z*). Si algún eje tiene ±1 G, entonces sabemos que ese eje está apuntando hacia abajo.

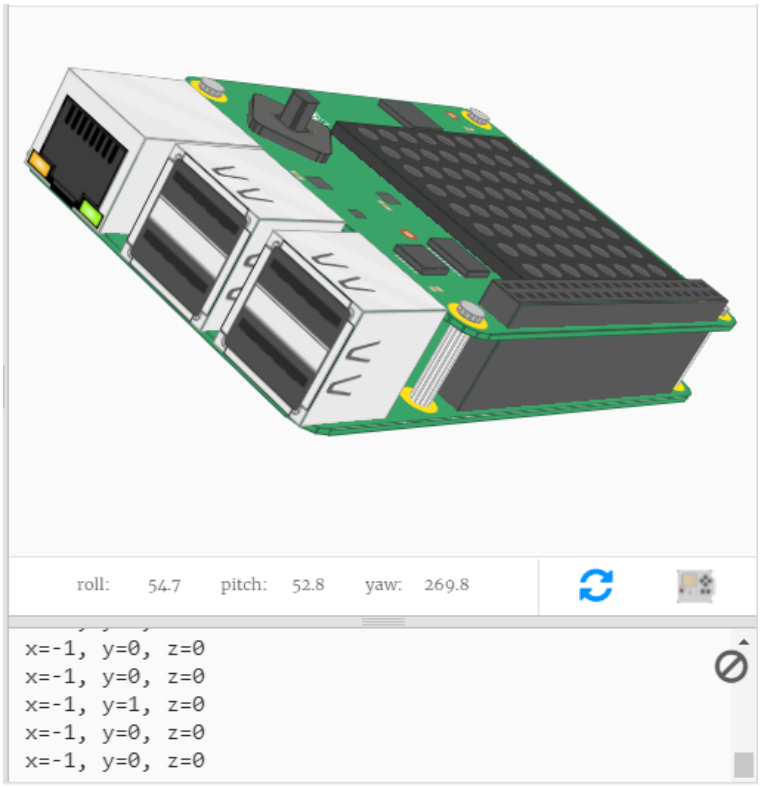
En este ejemplo, se mide la cantidad de aceleración gravitacional para cada eje (*x*, *y*, *z*) y se redondea el valor al número entero más cercano:

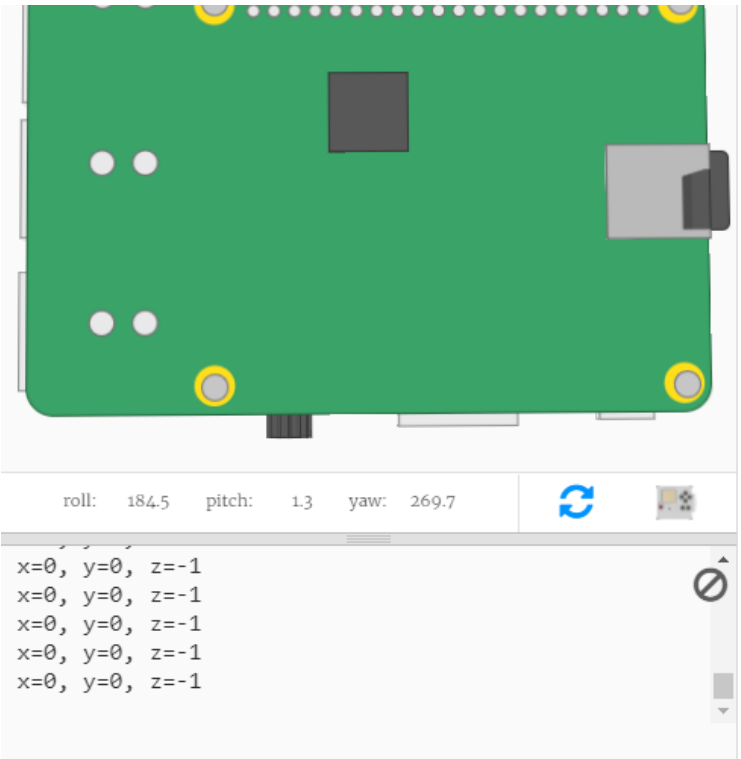
```
from sense_hat import SenseHat
sense = SenseHat()
while True:
    acceleration = sense.get_accelerometer_raw()
    x = acceleration['x']
    y = acceleration['y']
    z = acceleration['z']

    x=round(x, 0)
    y=round(y, 0)
    z=round(z, 0)
    print("x={0}, y={1}, z={2}".format(x, y, z))
```

3. Hacemos clic en **File** ('Archivo') **> Save As (**'Guardar como'), damos un nombre a nuestro programa; por ejemplo *acceleration.py*, después presionamos **F5** para ejecutar.

Cuando giramos la pantalla, deberíamos ver que los valores para *x* e *y* cambian entre -1 y 1. Si colocamos la computadora Pi apaisada o la giramos al revés, el eje z será 1 y después -1.

Algunas imágenes como ejemplos de los valores que se obtienen al mover la Raspberry Pi.


roll: 54.7 pitch: 52.8 yaw: 269.8
x=-1, y=0, z=0
x=-1, y=0, z=0
x=-1, y=1, z=0
x=-1, y=0, z=0
x=-1, y=0, z=0


roll: 184.5 pitch: 1.3 yaw: 269.7
x=0, y=0, z=-1
x=0, y=0, z=-1
x=0, y=0, z=-1
x=0, y=0, z=-1
x=0, y=0, z=-1

4. Si sabemos en qué posición está la Raspberry Pi, entonces podemos usar esa información para configurar la orientación de la matriz led.

Al del código anterior, le agregamos código antes del ciclo `while` para mostrar la letra *J* en la matriz les, usando el método `show_letter`, con el que ya estuvimos trabajando.

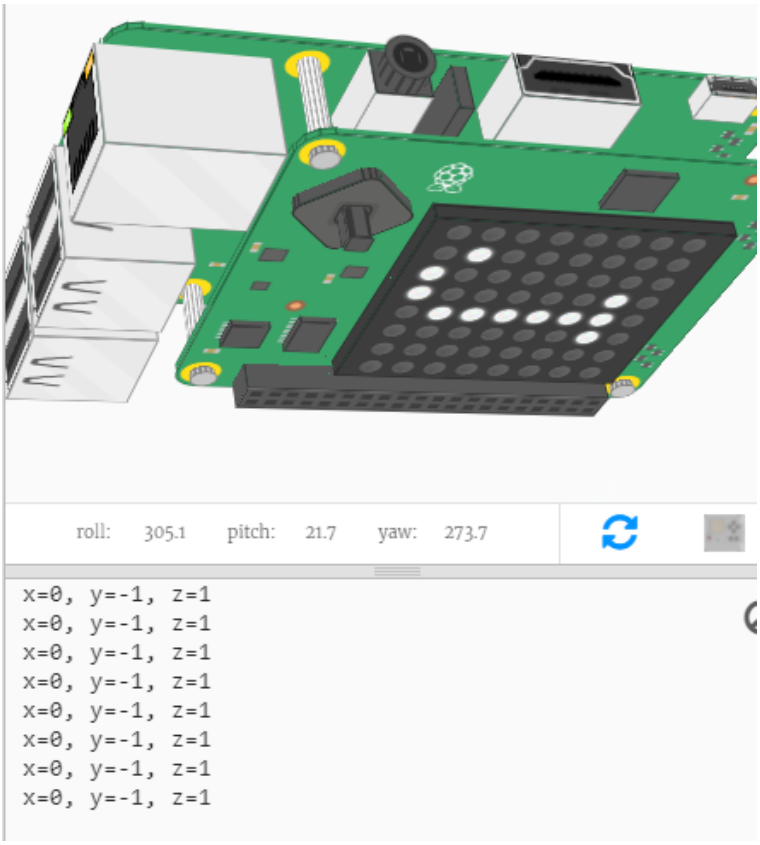
Después del código que muestra los valores de la fuerza G (gravedad) para los ejes *x*, *y* y *z*, hay que agregar una instrucción `if` para verificar en qué dirección apunta Sense HAT. Actualizamos la orientación de la pantalla utilizando el método `set_rotation`, que ya usamos en ejemplos anteriores.

Este es un pseudo código para que entendamos mejor las instrucciones `if`:

IF ('si') el eje **x tiene -1 G**, gire 180 grados
ELSE IF ('sino si') el eje **y tiene 1 G**, gire 90 grados
ELSE IF ('sino si') el eje **y tiene -1 G**, gire 270 grados
ELSE ('sino') rotar 0 grados

```
from sense_hat import SenseHat

sense = SenseHat()

# Display the letter J
sense.show_letter("J")

while True:
     acceleration = sense.get_accelerometer_raw()
     x = acceleration['x']
     y = acceleration['y']
     z = acceleration['z']

     x=round(x, 0)
     y=round(y, 0)
     z=round(z, 0)

     print("x={0}, y={1}, z={2}".format(x, y, z))

  # Actualización de la rotaciòn de la pantalla depen-
diendo hacia donde apunta la SENSE HAT
if x  == -1:
    sense.set_rotation(180)
  elif y == 1:
    sense.set_rotation(90)
  elif y == -1:
    sense.set_rotation(270)
  else:
    sense.set_rotation(0)
```

El siguiente es un ejemplo de los valores que se obtienen al orientar la Raspberry Pi como muestra la imagen:



5. Si la placa se gira, solo experimentará 1 G de aceleración en cualquier dirección; pero, al **sacudirla**, el sensor experimentaría más de 1 G. Entonces podríamos detectar ese movimiento rápido y darle una respuesta.

Para este programa, trabajaremos con la función `abs ()`, que no es específica de la biblioteca Sense HAT, sino que forma parte de Python estándar.

La función `abs ()` nos da el valor absoluto de un número; ignora si el valor real es positivo o negativo; por ejemplo, abs (1) y abs (-1) ambos devuelven 1. Esta función es útil porque no nos importa en qué dirección se está sacudiendo el sensor, solo que está siendo sacudido.

Podemos probarlo con este código:

```
from sense_hat import SenseHat

sense = SenseHat()

red = (255, 0, 0)

while True:
    acceleration = sense.get_accelerometer_raw()
    x = acceleration['x']
        y = acceleration['y']
        z = acceleration['z']

    x = abs(x)
    y = abs(y)
    z = abs(z)
el  tema detectar movimiento
    if x > 1 or y > 1 or z > 1:
        sense.show_letter("!", red)
    else:
        sense.clear()
```

Si al probar el programa notamos que es «demasiado sensible», es decir que el programa cree que la placa Sense HAT se agita constantemente, probemos cambiar el valor 1 por uno más grande para aumentar el umbral de lo que se define como *agitar*.