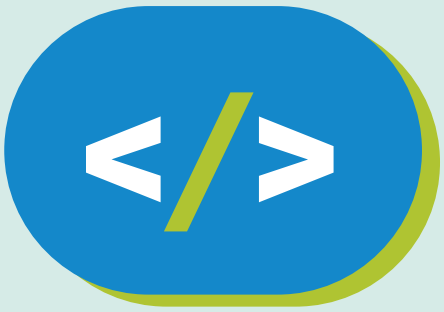


Kit de programación
Código Pi

Conocemos a 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

Subsecretario de Coordinación Administrativa
Javier Mezzamico

Directora Nacional de Innovación Educativa
María Florencia Ripani

ISBN en trámite

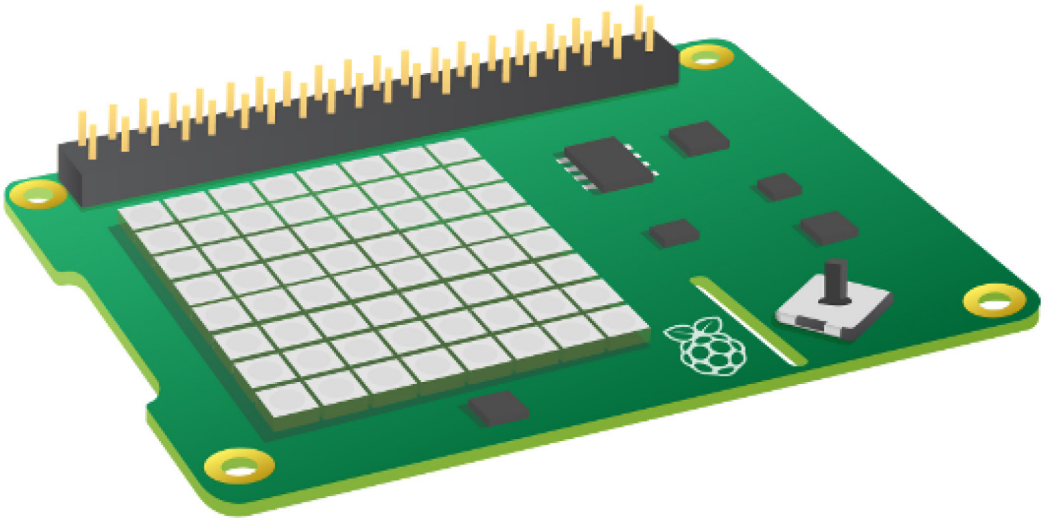
Índice

Conocemos a Sense Hat

Introducción

¿Qué vamos a hacer?

Conoceremos la placa Sense HAT que le permite a tu Raspberry Pi sensar al mundo a su alrededor.



¿Qué vamos a hacer?

En este proyecto, vas a aprender a controlar la pantalla de LED y a recolectar la información de los sensores del Sense HAT's LED, para luego combinar estas ideas en una serie de pequeños proyectos.

¿Qué vamos a aprender?

Siguiendo esta guía con tu Raspberry Pi y Sense HAT aprenderás a:

- Comunicarte con el Sense HAT usando Python
- Acceder a las salidas del Sense HAT
- Programar las entradas del Sense HAT
- Usar la librería Sense HAT para mostrar mensajes e imágenes
- Usar variables para almacenar información de los sensores
- Usar bucles para repetir acciones

¿Qué necesitás?

Hardware

- Raspberry Pi
- Sense HAT

Software

Necesitarás la última versión de Raspbian que ya incluye los siguientes paquetes de software:

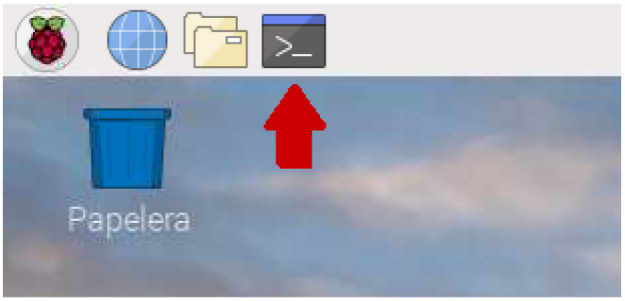
- Python 3
- Sense HAT para Python 3

Si por algún motivo necesitás instalar algún paquete de manera manual, seguí las siguientes instrucciones:

Instalar software en tu Raspberry Pi

Tu Raspberry Pi deberá estar conectado a internet para poder instalar paquetes de software. Antes de instalar cualquier paquete, actualizá a la última versión a Raspbian, el sistema operativo de tu Raspberry Pi.

- Para hacer esto, abrí una ventana de terminal e ingresá los siguientes comandos:




```
sudo apt-get update
sudo apt-get upgrade
```

- Ahora podés instalar los paquetes que necesitás por medio de comandos `install` en tu ventana de terminal. Por ejemplo, así se instala el software para Sense HAT:

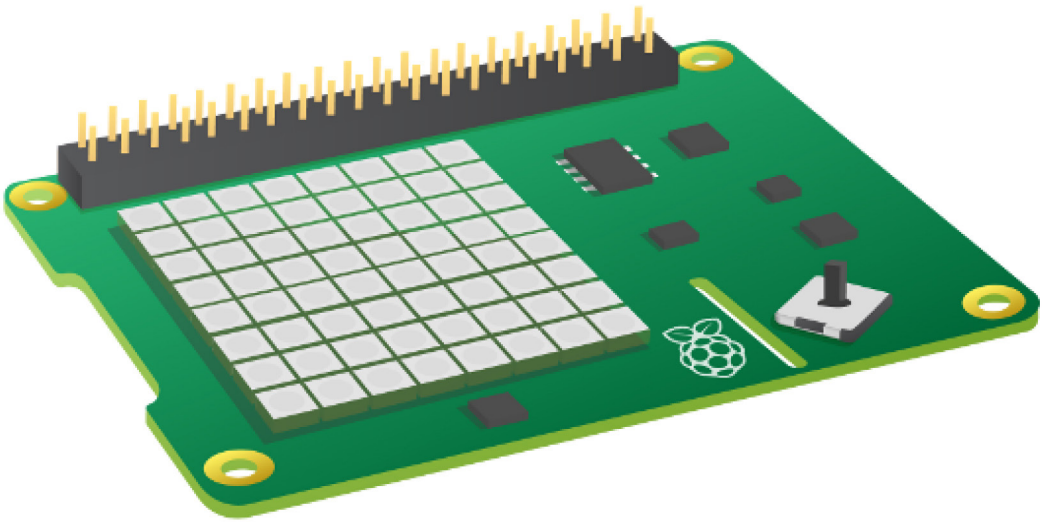
```
sudo apt-get install sense-hat
```

Escribí este comando en la terminal para instalar el paquete de Sense HAT:

```
sudo apt-get install sense-hat
```

¿Qué es un Sense HAT?

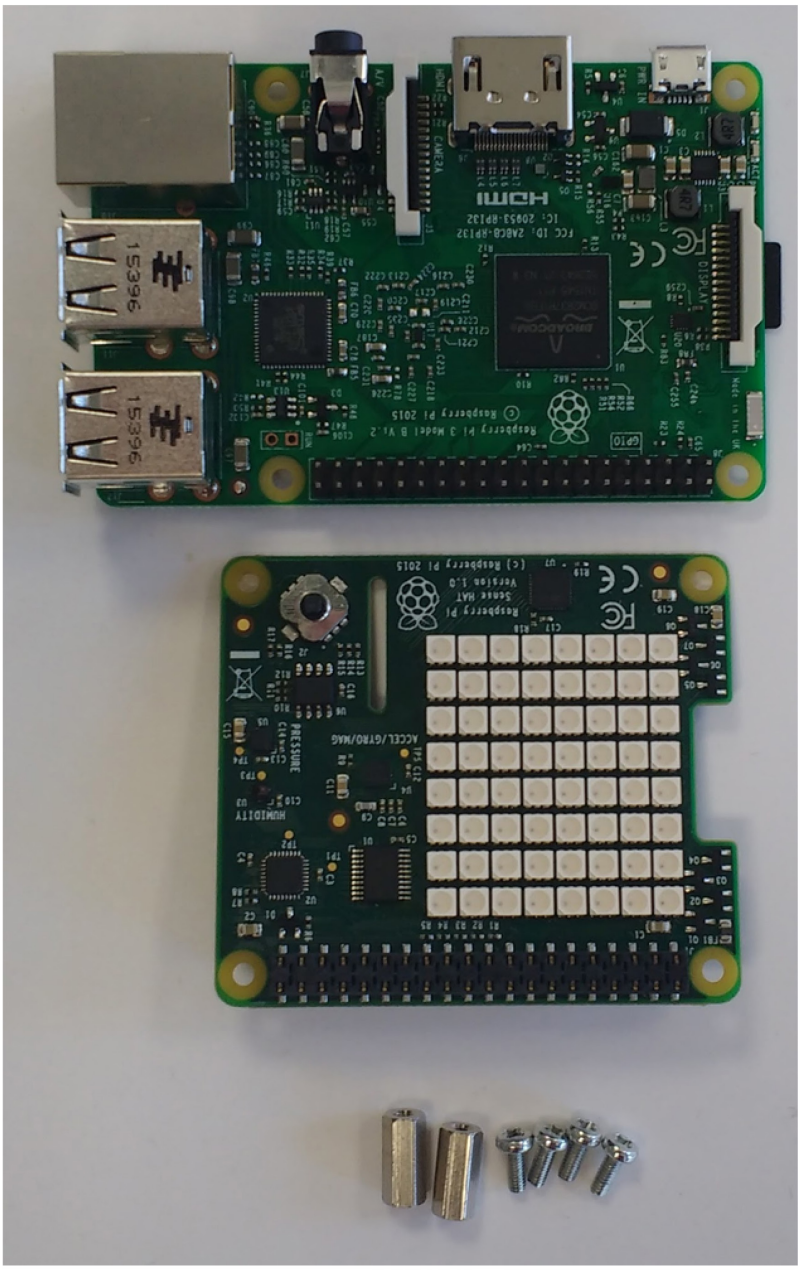
El Sense HAT es una placa adicional para tu Raspberry Pi que permite medir temperatura, humedad, presión y orientación, y mostrar información en su matriz de LED.



Conectando un Sense HAT

Antes de conectar cualquier HAT a tu Raspberry Pi, asegurate que esté apagado.

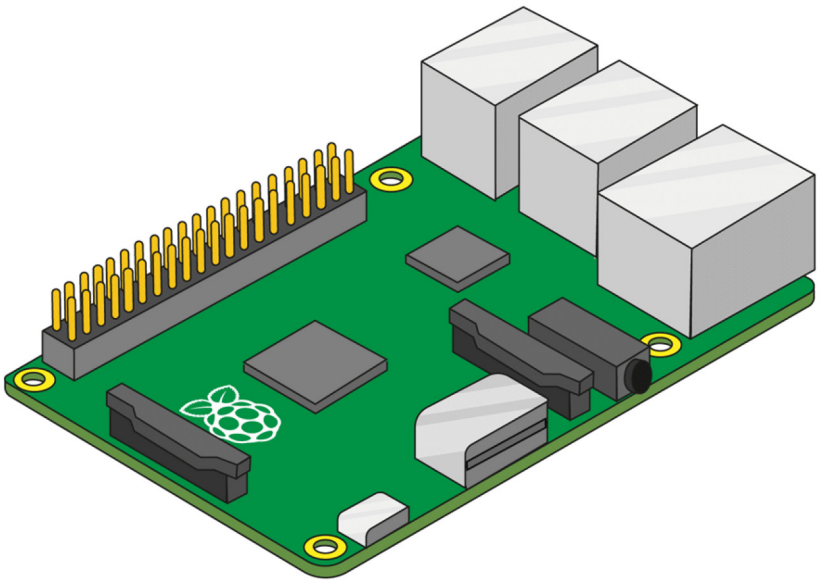
- Quitá el Sense HAT y sus partes del envoltorio.



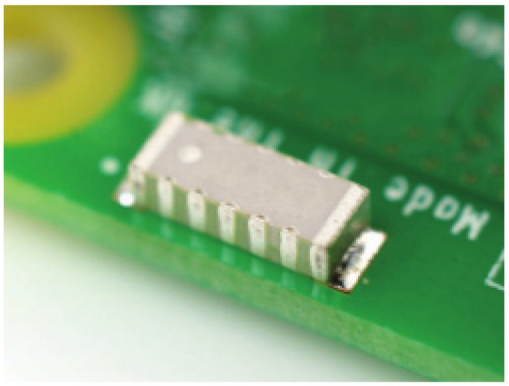
- Usá dos de los tornillos provistos para unir los espaciadores a tu Raspberry Pi, como se muestra más abajo.

Nota: el paso anterior es opcional — no es necesario agregar los espaciadores al Sense HAT para que funcione.

• Luego insertá el Sense HAT cuidadosamente en los pines de tu Raspberry Pi y aseguralo con los tornillos restantes.



Nota: usar un espaciador metálico cerca de la antena inalámbrica de tu Raspberry Pi 3 puede degradar su rendimiento o rango. Puedes omitir este espaciador, o usar espaciadores y tornillos de nylon.



Truco: tené cuidado cuando intentes desconectar tu Sense HAT, ya que el conector de 40 pines puede quedar enganchado.

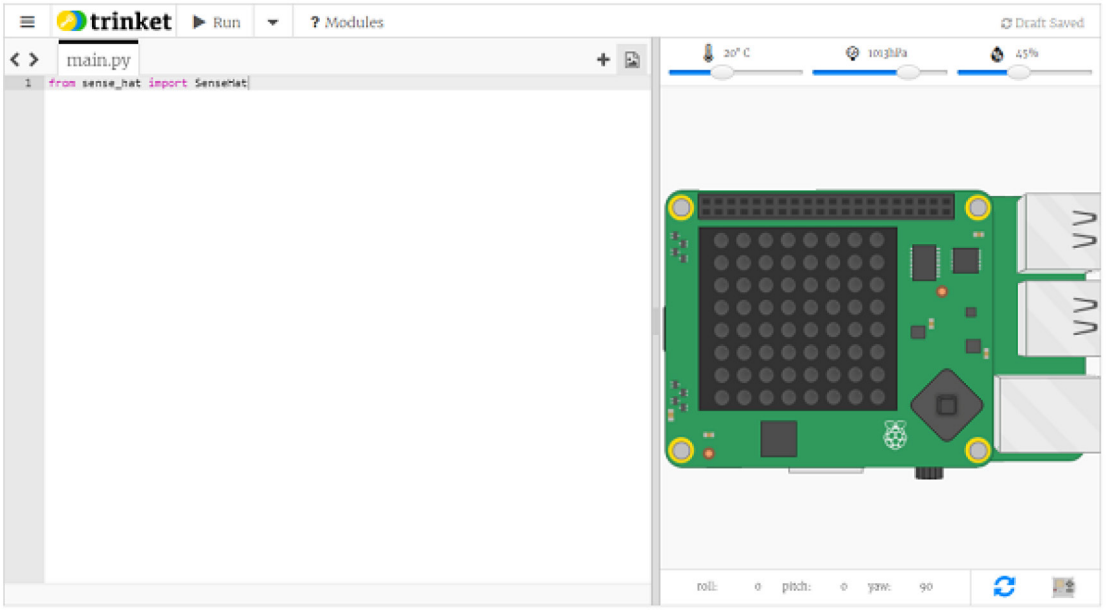
Si no tenés acceso a un Sense HAT real, podés usar un emulador.

Usando el emulador de Sense HAT

Si no tenés acceso a un Sense HAT real, podés usar un emulador.

Emulador online de Sense HAT

Hay un emulador online que podés usar en tu navegador para escribir y probar código para Sense HAT.

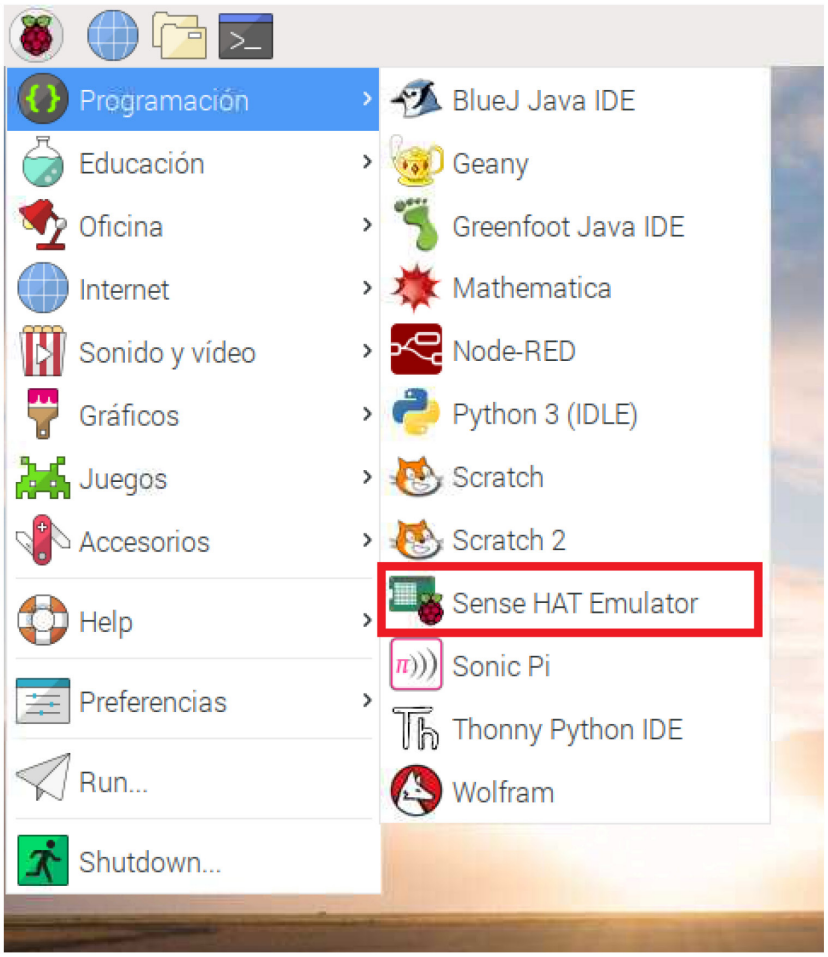


- Abrí un navegador de internet, entra a https://trinket.io/sense-hat y borrá el código de muestra del editor.

- Para poder guardar tu trabajo, tenés que crear una cuenta gratuita en el sitio web de Trinket.

Emulador de Sense HAT en tu Raspberry Pi

Tu Raspberry Pi incluye un emulador de Sense HAT en el sistema operativo Raspbian.



- Desde el menú principal, seleccioná Programación > Emulador Sense HAT para abrir una ventana que contiene al emulador.
- Si estás usando esta versión del emulador, tu programa debe importar desde `sense_emu` en vez de `sense_hat`:

```
from sense_emu import SenseHat
```

Si luego querés ejecutar tu código en un Sense HAT real, sólo tenés que cambiar la línea import como se ve debajo. El resto del código se mantiene exactamente igual.

```
from sense_hat import SenseHat
```

Mostrar texto

- Mostrá el texto “¡Hola mundo!” en la pantalla LED de tu Sense HAT.

Mostrar un mensaje en el Sense HAT

Asegurate que las siguientes líneas de código estén presentes en tu programa de Python, para configurar la conexión con el Sense HAT. Sólo es necesario agregarlas una vez.

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

- Agregá este código para mostrar un mensaje en la matriz de LED de tu Sense HAT.

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

El mensaje “Hola mundo” aparecerá en la pantalla de LED.

- Cambiá las palabras entre comillas (`""`) para ver un mensaje distinto.
Tu programa debería verse así:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.show_message("Hola mundo")
```

Podemos cambiar la manera de mostrar el mensaje agregando algunos **parámetros** extra al comando `show_message`.

`scroll_speed`: afecta la velocidad con la que el texto se mueve en la pantalla. El valor por defecto es `0.1`. Cuanto mayor el número, menor la velocidad.

`text_colour`: altera el color del texto y se define con tres valores, para especificar rojo, verde y azul. Estos valores se los conoce también como valores RGB.

Revisá las secciones de más abajo para aprender más acerca de los valores RGB.

Mostrar un color en el Sense HAT

- En un archivo de Python, escribí el siguiente programa:

```
from sense_hat import SenseHat

sense = SenseHat()

r = 255
g = 255
b = 255

sense.clear((r, g, b))
```

- Guardá y ejecutá tu código. La matriz LCD debería iluminarse de blanco brillante.

- Las variables `r`, `g`, y `b` representan los colores rojo, verde y azul respectivamente. Sus valores especifican qué tan brillante debería ser cada color, cada valor puede ir de `0` a `255`. En el programa de más arriba, el valor máximo de cada color ha sido usado, obteniendo el color blanco como resultado.

- También podés definir los tres valores RGB de un color usando una sola línea de código:

```
red = (255,0,0)
```

El resultado debería verse así:

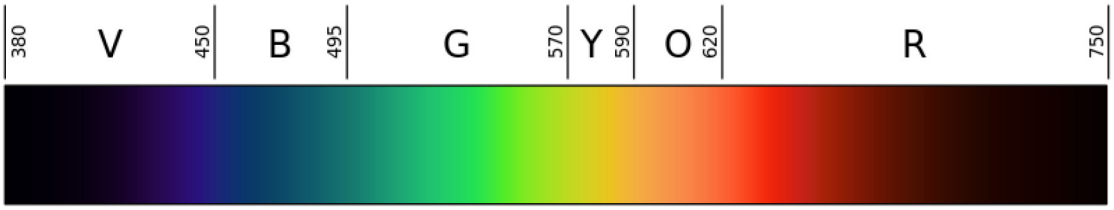
```
from sense_hat import SenseHat

sense = SenseHat()

r = 255
g = 255
b = 255

sense.clear((r, g, b))
```

- Cambiá el valor de uno de los colores, y corré el programa de nuevo. ¿Qué ves?

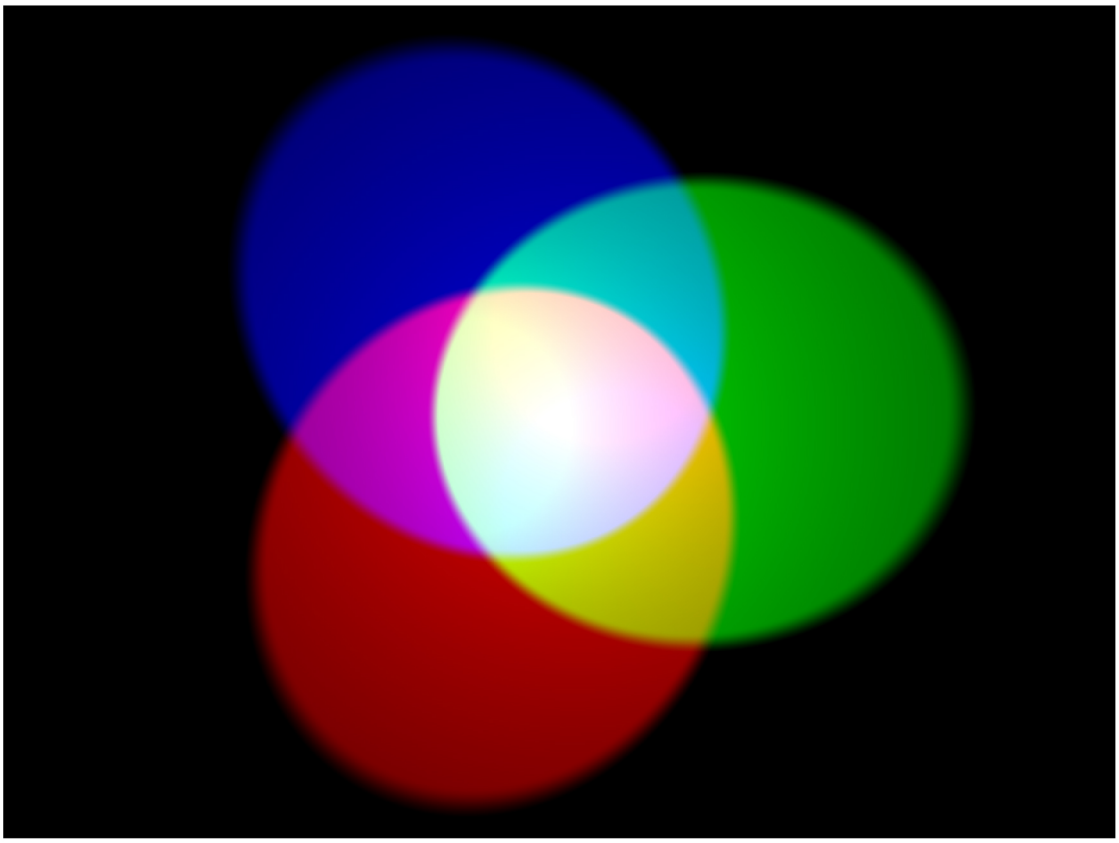
- ¿Qué otros colores podés hacer?

Representando colores con números

El color de un objeto depende del color de la luz que refleja o emite. La luz puede tener distintas longitudes de onda, y el color de la luz depende de su longitud de onda. El color de la luz de acuerdo a la longitud de onda puede verse en el diagrama de más abajo. Podés reconocer esto como los colores del arcoíris.



Los humanos vemos el color gracias a células especiales en nuestros ojos. Estas células se llaman *conos*. Tenemos tres tipos de conos, y cada tipo detecta luz roja, azul o verde. Por lo tanto todos los colores que vemos son simplemente mezclas de los colores rojo, azul y verde.



En la mezcla aditiva de colores, se usan tres colores primarios (rojo, verde y azul) para hacer los demás colores. En la imagen de arriba, hay tres reflectores de igual brillo, uno por cada color. en la ausencia de cualquier color, el resultado es negro. Si los tres colores son mezclados, el resultado es blanco. Cuando combinamos rojo y verde, el resultado es amarillo. Cuando combinamos rojo y azul, el resultado es magenta. Cuando combinamos azul y verde, el resultado es cian. Es posible hacer muchos más colores variando el brillo de los tres colores primarios.

Las computadoras almacenan todo en unos y ceros. Estos unos y ceros se organizan en conjuntos de ocho, llamados **bytes**.

Un byte puede representar cualquier valor de 0 a 255.

Cuando queremos representar un color en un programa informático, podemos hacerlo definiendo las cantidades de rojo, verde y azul que generan este color. Dichas cantidades se almacenan en un byte cada una y por lo tanto en números entre 0 y 255.

Aquí hay una tabla mostrando algunos valores de color:

Rojo	Verde	Azul	Color
255	00		Rojo
0	255	0	Verde
00		255	Azul
255	255	0A	marillo
255	0	255	Magenta
0	255	255	Cian

Podés encontrar un bonito selector de colores en w3schools.

`back_colour`: altera el color del fondo y funciona del mismo modo que `text_colour`.

• Agregá una línea de código antes de tu mensaje para definir una variable llamada `azul` con el valor `(0,0,255)`.

Creando una variable en Python

Una variable te permite almacenar información dentro de un programa. Las variables tienen un **nombre** y un **valor**.

Esta variable se llama `animal` y su valor es `gato`:

```
animal = "gato"
```

La siguiente variable se llama `puntaje` y su valor es `30`:

```
puntaje = 30
```

Para crear una variable, dale un nombre y asignale un valor. El nombre de la variable siempre va a la izquierda, por lo que el siguiente código es incorrecto:

```
# Este código es erróneo
30 = puntaje
```

• Agregá otra línea de código que defina una variable llamada `amarillo` con el valor `(255,255,0)`.

• Agregá parámetros en el comando `show_message` para mostrar el texto en amarillo con fondo azul.

Solución

La parte a agregar está resaltada en azul.

main.py

```
1  from sense_hat import SenseHat
2  sense = SenseHat()
3
4  azul = (0, 0, 255)
5  amarillo = (255, 255, 0)
6
7  sense.show_message("¡Astro Pi es increible!", text_colour=amarillo, back_colour=azul)
```

• Agregá otro parámetro llamado `scroll_speed` al comando `show_message` y fijá la velocidad en 0.05 `0.05` para acelerar el desplazamiento del mensaje.

• Poné el mensaje en un bucle while loop para que se repita.

Bucle While True en Python

El uso de un bucle while es el de repetir bloques de código una y otra vez siempre y cuando una condición sea `True`. Es por eso que los bucles while también son conocidos como repeticiones condicionales.

El ejemplo de aquí abajo es un bucle que se ejecutará por siempre - un bucle infinito o repetición incondicional. El bucle se ejecutará por siempre porque la condición es siempre `True`.

```
while True:
 print("¡Hola Mundo!")
```

Note: La línea de código `while`. declara la condición del bucle. La línea de código `print`. que se encuentra debajo está levemente desplazada hacia la derecha. Esto se llama indentación - la línea está indentada para mostrar que está dentro del bucle. Todo el código dentro del bucle será repetido.

Los bucles infinitos son útiles en situaciones donde queremos realizar las mismas acciones una y otra vez, por ejemplo revisar el valor de un sensor. Un bucle infinito como este va a continuar funcionando por siempre, lo que significa que cualquier línea de código escrita luego del bucle no se ejecutará nunca. Esto se conoce como **bloqueo** - porque el programa **bloquea** la ejecución del resto del código.

Solución

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

amarillo = (0, 0, 255)
azul = (255, 255, 0)

while True:
sense.show_message("¡Astro Pi es increíble!", text_colour=amarillo,
back_colour=azul, scroll_speed=0.05)
```

Mostrar un solo caracter

• Mostrá la letra “A” en el display de tu Sense HAT.

Mostrar una letra en el Sense HAT

Asegurate que las siguientes líneas de código estén presentes en tu programa de Python para configurar la conexión con el Sense HAT. Sólo es necesario agregarlas una vez.

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

- Agregá este código para mostrar la letra "Z" en la pantalla LED:

```
sense.show_letter("Z")
```

Si ejecutás este programa, la letra “Z” aparecerá en la pantalla. Podés cambiar la letra entre comillas (`""`) para variar la letra que aparece en pantalla.

Debería verse así:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.show_letter("Z")
```

Podemos cambiar la manera de mostrar la letra usando los dos mismos parámetros que utilizamos con el comando `show_message` : `text_colour` y `back_colour` . Las letras sueltas no se desplazan, por lo que no hay parámetro `scroll_speed` .

• Mostrá una letra “J” roja con fondo blanco.

• Usá la función `sleep` para mostrar las letras de tu nombre de a una a la vez, cada una con un color diferente, con un segundo de pausa entre cada una.

Usando el comando sleep de Python

Podés usar la función `sleep` para pausar temporalmente tu programa de Python.

- Agregá esta línea al principio de tu programa para importar la función `sleep`.

```
from time import sleep
```

- Cada vez que quieras una pausa en tu programa, llamá a la función sleep `sleep`. El número entre paréntesis indica cuántos segundos querés que dure la pausa.

```
sleep(2)
```

También podés pausar tu programa por fracciones de segundo.

```
sleep(0.5)
```

Solución

```
from sense_hat import SenseHat
from time import sleep

sense = SenseHat()

rojo = (255, 0, 0)
azul = (0, 0, 255)
verde = (0, 255, 0)
blanco = (255, 255, 255)
amarillo = (255, 255, 0)

sense.show_letter("L", rojo)
sleep(1)
sense.show_letter("a", azul)
sleep(1)
sense.show_letter("u", verde)
sleep(1)
sense.show_letter("r", blanco)
sleep(1)
sense.show_letter("a", amarillo)
```

• Generá un color aleatorio usando `randint` para elegir números entre `0` y `255` para cada uno de los tres valores RGB que componen un color.

Aleatoriedad en Python

Uno de los módulos standard en Python es el módulo `random`. Podés usarlo para crear números seudo-aleatorios en tu programa.

randint

Podés generar números enteros aleatorios entre dos valores usando la función `randint`. Por ejemplo, la siguiente línea de código produce un numero entero aleatorio entre 0 y 10 inclusive:

```
from random import randint
num = randint(0,10)
```

uniform

Si querés un número aleatorio racional, puedes usar la función `uniform`. Por ejemplo, la siguiente línea de código producirá un número racional aleatorio mayor o igual a 0, pero menor a 10.

```
from random import uniform
num = uniform(0,10)
```

choice

Si querés seleccionar un elemento aleatorio de una lista, puedes usar la función `choice`.

```
from random import choice
baraja = ['Oro', 'Copa', 'Espada', 'Basto']
card = choice(baraja)
```

Solución

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

sense = SenseHat()

rojo = (255, 0, 0)
azul = (0, 0, 255)
verde = (0, 255, 0)
blanco = (255, 255, 255)
amarillo = (255, 255, 0)

# Generar un color aleatorio

def pick_random_colour():
random_rojo = randint(0, 255)
random_verde = randint(0, 255)
random_azul = randint(0, 255)
return (random_rojo, random_verde, random_azul)

sense.show_letter("L", pick_random_colour())
sleep(1)
sense.show_letter("a", pick_random_colour())
sleep(1)
sense.show_letter("u", pick_random_colour())
sleep(1)
sense.show_letter("r", pick_random_colour())
sleep(1)
sense.show_letter("a", pick_random_colour())
sleep(1)

sense.clear()
```

- Utilizá `sense.clear()` al final de tu programa para borrar la pantalla LED.

Mostrar imágenes

Podés llenar toda la pantalla LED con un color determinado, utilizando el método `clear` con un color que elijas.

Mostrando un color en el Sense HAT

```
from sense_hat import SenseHat

sense = SenseHat()

r = 255
g = 255
b = 255

sense.clear((r, g, b))
```

- Guardá y ejecutá tu código. La matriz LCD debería iluminarse de blanco brillante.
- Las variables `r`, `g`, y `b` representan los colores rojo, verde y azul respectivamente. Sus valores especifican qué tan brillante debería ser cada color, cada valor puede ir de `0` a `255`. En el programa de más arriba, el valor máximo de cada color ha sido usado, obteniendo el color blanco como resultado.
- También podés definir los tres valores RGB de un color usando una sola línea de código:

```
red = (255,0,0)

from sense_hat import SenseHat

sense = SenseHat()

r = 255
g = 255
b = 255

sense.clear((r, g, b))
```

• Cambiá el valor de uno de los colores, y ejecutá el programa de nuevo. ¿Qué ves?

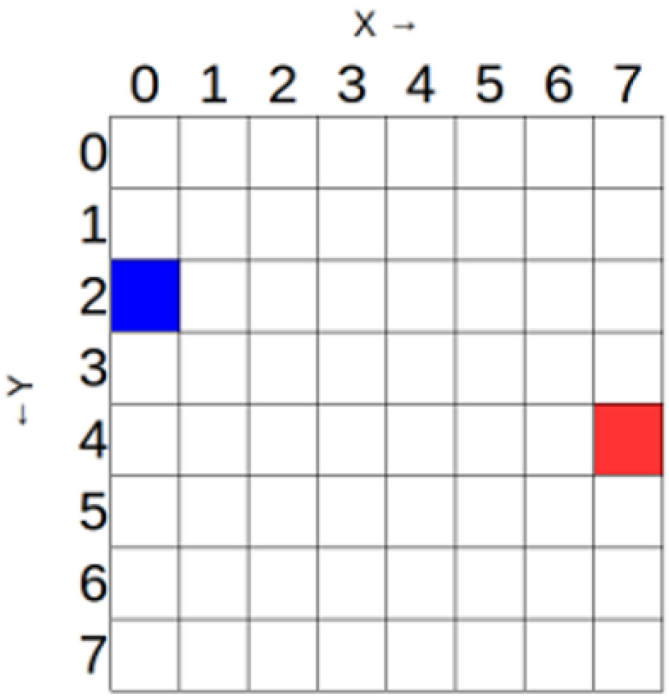
• ¿Qué otros colores puedes hacer?

Encendiendo píxeles individuales

¡La pantalla LCD puede mostrar más que sólo texto! Podemos controlar cada LED individualmente para crear una imagen.

Coordenadas de la pantalla LCD del Sense HAT

La pantalla LED del Sense HAT usa un sistema de coordenadas con un eje X y un eje Y. La numeración de ambos ejes inicia en 0 (no en 1) en la esquina superior izquierda. Cada LED puede ser usado como un pixel de una imagen, y puede ser referenciado usando la notación `x,y`.



El pixel azul está en las coordenadas `0, 2`. El pixel rojo está en las coordenadas `7, 4`.
Podés encender los pixeles (LEDs) individualmente utilizando el método `set_pixel()`.

Para replicar el diagrama de más arriba, podemos escribir un programa como este:

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

azul = (0, 0, 255)
rojo = (255, 0, 0)

sense.set_pixel(0, 2, azul)
sense.set_pixel(7, 4, rojo)
```

• Encendé los píxeles (LEDs) de las cuatro esquinas de la pantalla, del color que escojas.

¿Podés adivinar lo que mostrará el siguiente programa?

```
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))
```

• Cambiá el código para mostrar una imagen diferente.

Encendiendo múltiples pixeles

Encender pixeles individuales funciona muy bien, pero se vuelve muy complejo cuando queremos encender múltiples pixeles. Para cambiar todos los pixeles en un comando, podemos utilizar `set_pixel()`.

- Usá el método `set_pixel()` para mostrar una imagen en la pantalla LED.

Encendiendo múltiples pixeles en el Sense HAT

Es tentador intentar dibujar formas en la pantalla de tu Sense HAT utilizando el comando `set_pixel()` una y otra vez. ¡Sin embargo, existe un comando `set_pixel()` que nos permite cambiar los 64 pixeles en un solo comando! Por ejemplo, podemos dibujar la cara de un creeper de Minecraft así:

```
from sense_hat import SenseHat

sense = SenseHat()

# Definir algunos colores

g = (0, 255, 0) # Verde
b = (0, 0, 0) # Negro

# Definimos donde mostrar cada color

creeper_pixels = [
 g, g, g, g, g, g, g, g,
 g, g, g, g, g, g, g, g,
 g, b, b, g, g, b, b, g,
 g, b, b, g, g, b, b, g,
 g, g, g, b, b, g, g, g,
 g, g, b, b, b, b, g, g,
 g, g, b, b, b, b, g, g,
 g, g, b, g, g, b, g, g
]

# Mostrar esos colores en la pantalla LED

sense.set_pixels(creeper_pixels)
```

Inclusive podemos usar más de dos colores, como en este ejemplo de Steve de Minecraft:

```
from sense_hat import SenseHat

sense = SenseHat()

# Definir algunos colores

B = (102, 51, 0)
b = (0, 0, 255)
S = (205,133,63)
W = (255, 255, 255)

# Definir donde mostrar cada color

steve_pixels = [
B, B, B, B, B, B, B, B,
B, B, B, B, B, B, B, B,
B, S, S, S, S, S, S, B,
  S, S, S, S, S, S, S, S,
  S, W, b, S, S, b, W, S,
  S, S, S, B, B, S, S, S,
  S, S, B, S, S, B, S, S,
  S, S, B, B, B, B, S, S
]

# Mostrar esos colores en la pantalla LED

sense.set_pixels(steve_pixels)
```

Cambiando la orientación

Hasta ahora, todos nuestros textos e imágenes aparecieron con la misma orientación, con el puerto HDMI abajo. Sin embargo, quizá no sea esta la orientación real del Sense HAT (especialmente en el espacio exterior), por lo que a veces quizá quieras cambiar la orientación de la pantalla LED.

Rotando la pantalla LED del Sense HAT

Podés cambiar la orientación de la pantalla LED de tu Sense HAT. Utilizá el método `set_rotation()` para rotar la pantalla de cuatro maneras: 0, 90, 180, o 270 grados.

- Por ejemplo, para rotar la pantalla 180 grados podés usar la siguiente línea de código:

```
sense.set_rotation(180)
```

- También podés invertir la imagen en la pantalla, ya sea horizontalmente...

```
sense.flip_h()
```

... o verticalmente.

```
sense.flip_v()
```

• Cambiá la orientación de la imagen de píxeles del paso anterior agregando código para rotar la pantalla debajo del código que escribiste para conectar a tu Sense HAT:

```
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))
```

Podés crear una animación simple invirtiendo una imagen repetidamente:

```
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()
```

Sensando el ambiente

El Sense HAT tiene una serie de sensores ambientales para detectar las condiciones que lo rodean; puede medir presión, temperatura y humedad.

Leyendo la presión con el Sense HAT

- En un archivo de Python, ingresá el siguiente programa:

```
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()

pressure = sense.get_pressure()
print(pressure)
```

- Cuando ejecutes el programa, deberías ver algo así:

```
1013.40380859
```

Detectando la temperatura con el Sense HAT

Tu Sense HAT tiene dos sensores con capacidad de leer la temperatura ambiente: el sensor de humedad y el sensor de presión. `get_temperature_from_humidity` lee la temperatura desde el sensor de humedad (`get_temperature` es una versión corta del mismo comando). `get_temperature_from_pressure` lee la temperatura desde el sensor de presión.

- En un archivo de Python, escribí el siguiente programa:

```
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()

temp = sense.get_temperature()
print(temp)
```

- Deberías ver algo así:

```
28.6293258667
```

Detectando humedad con tu Sense HAT

- En un archivo de Python, escribí el siguiente programa:

```
from sense_hat import SenseHat

sense = SenseHat()
sense.clear()

humidity = sense.get_humidity() # obtener humedad del sensor
print(humidity) # imprimir humedad
```

- Cuando ejecutes el programa, deberías ver algo así:

```
34.6234588623
```

• Creá un texto que mantenga a la gente informada acerca de la temperatura, presión y humedad ambiente. Podés utilizar el código de desplazamiento de texto que creaste en el paso ‘Mostrar texto’ como ayuda.

Solución

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

while True:

# Toma lecturas de los tres sensores

t = sense.get_temperature() # temperatura
p = sense.get_pressure() # presión
h = sense.get_humidity() # humedad

# Redondea los valores a un lugar decimal

t = round(t, 1)
p = round(p, 1)
h = round(h, 1)

# Crea el mensaje
# str() convierte el valor en una cadena de caracteres, para que
pueda ser concatenado

message = "Temperatura: " + str(t) + " Presión: " + str(p) + "
Humedad: " + str(h)

# Muestra el mensaje

sense.show_message(message, scroll_speed=0.05)
```

De acuerdo a la documentación en línea, la Estación Espacial Internacional mantiene estas condiciones en los siguientes niveles:

Temperatura: 18.3-26.7 Celsius

Presión: 979-1027 milibares

Humedad: aproximadamente 60%

• Definí variables para los colores verde (0, 255, 0) y rojo (255, 0, 0).
• Usá un condicional `if` en tu programa para corroborar si la temperatura se encuentra entre 18,3 y 26,7 grados Celsius.

Operadores Booleanos en condicionales de Python

- Una alternativa condicional `if` en Python evalúa una sola condición. Por ejemplo:

```
x = 5
if x > 5:
     print('x es mayor a cero')
```

- Pero, a veces quizá quieras evaluar por más de una condición. En esos casos, podés usar operadores lógicos en tu programa.
- El operador de conjunción `and` evalúa si ambas condiciones son ciertas simultáneamente (intersección lógica). Por ejemplo:

```
x = 5
if x > 0 and x < 10:
 print('x está entre 0 y 10')
```

- Siempre que `x` sea cualquier número dentro del grupo - `1,2,3,4,5,6,7,8,9`, la condición será cierta.
- Podés usar el operador de disyunción `or`, que evalúa si alguna de las condiciones (o ambas) es cierta (suma lógica).

```
x = 5
if x > 0 or x < 10:
     print('x existe')
```

- En este caso, la condición será verdadera siempre que `x` sea mayor que `0`, o menor a `10`.

• Si la temperatura está dentro del rango normal, mostraremos un mensaje con un fondo verde. Sino, mostraremos un fondo rojo.

Solución

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

# Define los colores rojo y verde

rojo = (255, 0, 0)
verde = (0, 255, 0)

while True:

# Toma lecturas de los tres sensores

t = sense.get_temperature() # temperatura
p = sense.get_pressure() # presión
h = sense.get_humidity() # humedad

# Redondea los valores a un lugar decimal

t = round(t, 1)
p = round(p, 1)
h = round(h, 1)

# Crea el mensaje
# str() convierte el valor en una cadena de caracteres, para que
pueda ser concatenado

message = "Temperatura: " + str(t) + " Presión: " + str(p) + "
Humedad: " + str(h)

if t > 18.3 and t < 26.7:
 bg = verde
else:
 bg = rojo

# Muestra el mensaje

sense.show_message(message, scroll_speed=0.05, back_colour=bg)
```

• Agregá más condicionales `if` para evaluar las condiciones normales de presión y humedad.

Detectando movimiento

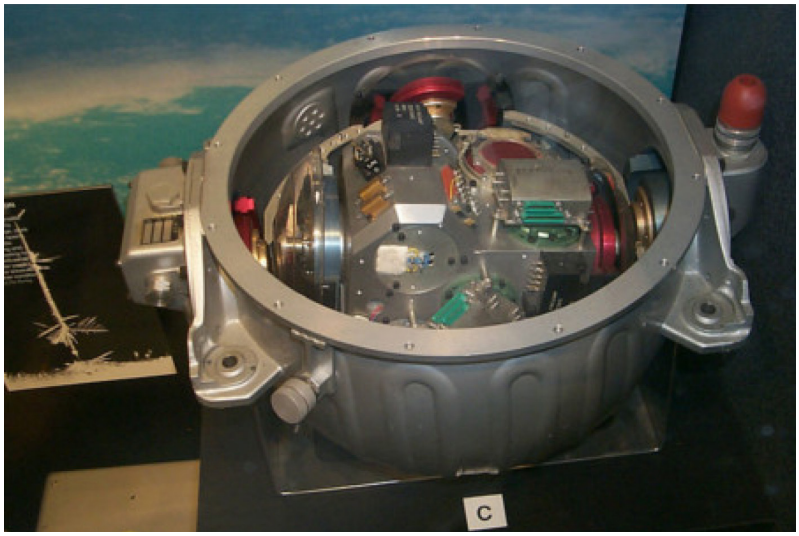
El Sense HAT tiene un chip IMU (Unidad de Medición Inercial) que incluye un conjunto de sensores que detectan movimiento:

- Un giróscopo (para detectar la orientación de la placa)
- Un acelerómetro (para detectar movimiento)
- Un magnetómetro (para detectar campos magnéticos)

¿Qué es un IMU?

El Sense HAT tiene un sensor de movimiento llamado, IMU, que mide varios tipos de moviento. IMU quiere decir Unidad de Medición Inercial. En realidad, son tres sensores en uno:

- Giróscopo: mide inercia y rotación
- Acelerómetro: mide fuerzas de aceleración, puede ser usado para detectar la dirección de la gravedad
- Magnetómetro: mide el campo magnético de la Tierra (similar a una brújula)



¿Por qué es importante un sensor de movimiento? Cuando estás en el espacio, hay una pregunta de importancia fundamental, de la que siempre tienes que saber la respuesta: "¿Hacia dónde estoy apuntando?"

Si no sabes tu orientación, estás en graves problemas. Entonces los sensores IMU como el de tu Sense HAT se usan en naves espaciales tripuladas y no tripuladas para seguir los movimientos y mantener un entendimiento de la orientación.

Inclusive las primeras naves espaciales tenían sensores IMU - preguntale a tus abuelos si recuerdan la misión Apolo que llevó humanos a la superficie de la Luna.

Arriba hay una fotografía del sensor IMU del módulo de comando Apolo. Notarás qué grande es comparado con el diminuto cubo negro en el Sense HAT - esa es la diferencia entre tecnología de 1975 y 2015. Incidentalmente, el IMU del Sense HAT probablemente no sea tan exacto como el del Apolo; ¡sin embargo, es un millón de veces más económico!

Aprendiendo sobre elevación, alabeo y dirección

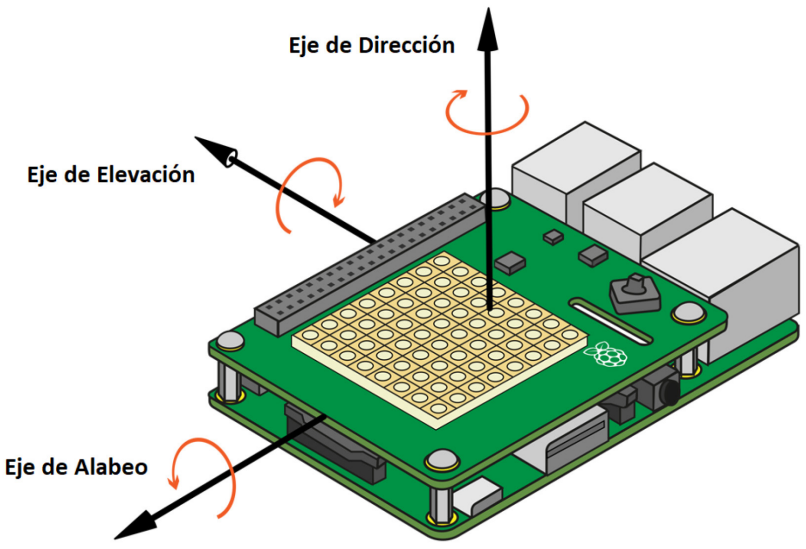
Todos sabemos que la Tierra rota alrededor de un eje que va del Polo Norte al Polo Sur. Todos los objetos tienen tres ejes alrededor de los que pueden rotar:

- **Elevación** — imaginá un avión despegando
- **Alabeo** — imaginá un avión haciendo un "tirabuzón"
- **Dirección** — imaginá a un avión cambiando de dirección como un automóvil

Si sabes cuánta rotación ocurrió en cada eje de un objeto, entonces sabes en qué dirección apunta dicho objeto.

Mirá este video que muestra estos tres ejes en relación a un avión. Imaginá un avión apuntando en una dirección aleatoria en el espacio. Para fijar el avión en dicha dirección, puedes rotarlo una cantidad determinada en cada eje.

La siguiente imagen muestra donde están esos tres ejes en relación con el Sense HAT.



• Escribí un programa que detecte la elevación, alabeo, y dirección. Ejecutá el programa y mueve el Sense HAT. Mira como los valores cambian con el movimiento del Sense HAT.

Detectando elevación, alabeo y dirección con el Sense HAT

El Sense HAT tiene sensores de orientación que detectan elevación, alabeo y dirección. Haz lo siguiente para acceder a esa información.

- En un archivo de Python, escribí el siguiente programa:

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

o = sense.get_orientation() # obtener orientacion del IMU
pitch = o["pitch"] # obtener elevación
roll = o["roll"] # obtener alabeo
yaw = o["yaw"] # obtener dirección
print("pitch {0} roll {1} yaw {2}".format(pitch, roll, yaw)) #
imprimir el resultado
```

- Cuando ejecutes el programa, deberías ver algo así:

```
pitch 356.35723002363454 roll 303.4986602798494 yaw 339.19880231669873
```

Nota: Cuando usás los sensores de movimiento, es importante tomar lecturas bastante seguido. Si tomas lecturas muy poco seguido, por ejemplo agregando `time.sleep(0.5)` en tu bucle, verás resultados extraños. Esto es porque el código necesita muchas mediciones para poder combinar de manera exitosa la información proveniente del giróscopo, acelerómetro y magnetómetro.

¿Hacia dónde es arriba?

El método `sense.get_accelerometer_raw()` informa la cantidad de fuerza-G activa en cada eje (x, y, z). si algún eje indica ±1G, entonces sabemos que ese eje está apuntando hacia abajo.

En este ejemplo, la cantidad de aceleración gravitacional de cada eje es extraída y luego redondeada al número entero más próximo:

```
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))
```

- Rotá el Sense HAT. Deberías ver los valores de `x` e `y` cambiar entre `-1` y `1` . Si ponés tu Raspi horizontal o boca abajo, el valor del eje `z` será `1` o de `-1` respectivamente.

Usá esta información para cambiar la orientación de la pantalla LED.

- Empezando con el programa de más arriba, agregá un poco más de código antes del bucle `while` para mostrar la letra “J” en la pantalla LED. Usa el método `show_letter` que ya conocés.

- Luego del código que muestra los valores de fuerza-G para los ejes x, y, z, agregá un condicional `if` o `if-else` que revise en qué orientación se encuentra el Sense HAT. Luego, actualizá la orientación de la pantalla usando el método `set_rotation` que ya conocés. Aquí hay algo de pseudo-código para empezar:

`If` el eje `x` tiene `-1` G, `rotate` `180` grados `Else if` el eje `y` tiene `1` G, `rotate` `90` grados `Else if` el eje y tiene `-1` G, `rotate` `270` grados `Else` `rotate` `0` grados

Solución

```
from sense_hat import SenseHat

sense = SenseHat()

# Mostrar la letra 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))

# Actualizar la orientación de la pantalla de acuerdo a la dirección
del 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)
```

t

Sacudí la placa

Si sólo rotamos la placa, obtendremos solamente 1G de aceleración en cualquier dirección; si la sacudimos, el sensor detectará más de 1G. Podemos así detectar movimientos rápidos y responder.

Para este programa usaremos la función `abs()`, que no es específica del Sense HAT sino parte estándar de Python. `abs()` devuelve el valor numérico absoluto de un valor e ignora si dicho valor es positivo o negativo - por ejemplo, `abs(1)` y `abs(-1)` ambas devuelven 1. Esta función es útil porque no nos interesa saber en qué dirección es agitado el sensor, sino simplemente si es agitado.

```
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)

    if x > 1 or y > 1 or z > 1:
        sense.show_letter("!", red)
    else:
        sense.clear()
```

Esto es un poco difícil de emular, así que deberías intentar hacer esto con un verdadero Sense HAT. Si encontrás que tu programa es muy sensible (esto es que

tu programa piense que el Sense HAT está siendo agitado continuamente), podés intentar cambiar el valor `1` por uno más grande para subir el umbral de lo que está definido como "agitar".

Usando el joystick

Podés detectar cuando el joystick del Sense HAT es pulsado, mantenido y soltado en cinco direcciones diferentes: arriba, abajo, izquierda, derecha, y medio.

Detectando movimiento del joystick con el Sense HAT

El joystick de tu Sense HAT está asignado al cursor de tu teclado, y su pulsación central a la tecla `Return/Enter`. Esto significa que usar el joystick es exactamente lo mismo a presionar esas teclas en el teclado. Recordá que la dirección hacia abajo es con el puerto HDMI mirando hacia abajo.

- En un archivo de Python, ingresá el siguiente programa:

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


while True:
    for event in sense.stick.get_events():
        print(event.direction, event.action)
```

Este programa escribirá en pantalla la dirección en la que es pulsada el joystick, o la dirección en la que se suelta.

- Cuando ejecutes el programa y muevas el joystick en varias direcciones, deberías ver algo así en la ventana terminal. En el emulador, podés simular movimiento del joystick usando las flechas de cursor del teclado.

• Dependiendo la dirección en la que pulses el teclado, mostrar las letras U, D, L, R o M en la pantalla LED.

```
('up', 'pressed')
('up', 'released')
('down', 'pressed')
('down', 'released')
('left', 'pressed')
('left', 'released')
('right', 'pressed')
('right', 'released')
('middle', 'pressed')
('middle', 'released')
```

Solución

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

e = (0, 0, 0)
w = (255, 255, 255)

sense.clear()
while True:
for event in sense.stick.get_events():
 # Revisa si el joystick es pulsado
 if event.action == "pressed":


# Revisa la dirección

if event.direction == "up":
 sense.show_letter("U")      # Flecha arriba
elif event.direction == "down":
 sense.show_letter("D")      # Flecha abajo
elif event.direction == "left":
 sense.show_letter("L")      # Flecha izquierda
elif event.direction == "right":
 sense.show_letter("R")      # Flecha derecha
elif event.direction == "middle":
 sense.show_letter("M")      # Tecla enter

# Espera un poco y limpia la pantalla

sleep(0.5)
sense.clear()
```

También podés llamar una función cada vez que el joystick de tu Sense HAT es movido en una dirección particular.

Disparando llamadas a funciones con el joystick de Sense HAT

El joystick de Sense HAT puede ser usado para disparar llamadas a funciones en respuesta a su movimiento.

- Por ejemplo, podés decir a tu programa que "escuche" continuamente a un evento específico, como el joystick siendo pulsado hacia arriba (`direction_up`), y disparar una función (llamada `pushed_up` en este ejemplo) en respuesta.

```
sense.stick.direction_up = pushed_up
```

- La función disparada por el evento puede no tener parámetros, o puede tomar el evento como parámetro. En el ejemplo de abajo, el evento simplemente es mostrado en pantalla.

```
def pushed_up(event):
  print(event)
```

- Esta función mostrará el tiempo del evento, la dirección en la que fue movida el joystick, y la acción específica. La salida se verá así:

```
InputEvent(timestamp=1503565327.399252,direction=u'up',action=u'pressed')
```

- Otro ejemplo útil es el método `direction_any`:

```
sense.stick.direction_any = do_thing
```

• Si usas ese evento como vimos en el ejemplo, la función `do_thing` será disparada en respuesta a cualquier evento del joystick. Por ejemplo, podés definir la función `do_thing` para que reporte el evento en inglés.

```
def do_thing(event):
    if event.action == 'pressed':
        print('You pressed me')
        if event.direction == 'up':
            print('Up')
        elif event.direction == 'down':
            print('Down')
    elif event.action == 'released':
        print('You released me')
```

•Creá funciones para llenar la pantalla LED con cuatro colores diferentes. Agregá disparadores para llamar una función diferente para cada dirección posible del joystick.

Solución

```
from sense_hat import SenseHat

sense = SenseHat()

# Define las funciones

def red(): # rojo
sense.clear(255, 0, 0)

def blue(): # azul
sense.clear(0, 0, 255)

def green(): # verde
sense.clear(0, 255, 0)

def yellow(): # amarillo
sense.clear(255, 255, 0)

# Asocia cada funcion a cada direccion

sense.stick.direction_up = red
sense.stick.direction_down = blue
sense.stick.direction_left = green
sense.stick.direction_right = yellow
sense.stick.direction_middle = sense.clear    # Presiona la tecla
enter

while True:
pass  # Esto mantiene el programa funcionando para recibir eventos
de joystick
```

Desafío: poniendo todo junto

Ahora que exploraste la mayoría de las características de tu Sense HAT, podés combinarlas para crear un proyecto. Debajo hay un ejemplo de un juego de reacción.

El juego consiste en rotar la placa para hacer que la flecha apunte hacia arriba. Si

logras hacerlo a tiempo, la flecha se pondrá verde y tu puntaje aumentará; sino, la flecha se pondrá roja y el juego termina. El juego sigue mostrando flechas en nuevas orientaciones hasta que pierdas, y cada vez lo hace más rápido.

Esta idea combina:

- Mostrar mensajes e imágenes en la pantalla LED
- Fijar y detectar la orientación
- Uso de variables, aleatoriedad, iteración y selección

Como este programa es más complicado que los anteriores en esta guía, es conveniente planear los pasos involucrados en pseudocódigo:

`Import` las librerías requeridas (`sense_hat` , `time` , `random`)
Crea el objeto `sensehat`

Define variables para cada color (blanco, negro, verde, rojo)

Crea las tres flechas (blanco, verde, rojo)
Fija una variable `pause` a 3 (tiempo inicial entre flechas)
Fija las variables `score` y `angle` a `0`
Fija una variable llamada `play` a `True` (será usada luego para iniciar y detener el juego)

```
while play == True
```

Selecciona un ángulo `random` Muestra una flecha blanca `Sleep` por el tiempo de pausa de la variable `pause`
`If` la orientación es igual a la flecha...
Agrega un punto a `score` y cambia la flecha a verde
Caso contrario fija la variable `play` a `False` y muestra la flecha roja

Acorta la duración de `pause` levemente
Pausa antes de la próxima flecha

Al salir del bucle, mostrar un mensaje con el `score`

Solución

```
# Importa las librerías necesarias (sense_hat, time, random)

from sense_hat import SenseHat
from time import sleep
from random import choice

# Crea un objeto sense

sense = SenseHat()

# Crea los colores (blanco, verde, rojo, negro)

w = (150, 150, 150) # blanco
g = (0, 255, 0) # verde
r = (255, 0, 0) # rojo
e = (0, 0, 0) # negro

# Crea imagenes para las tres flechas coloreadas

arrow = [ # flecha blanca
e,e,e,w,w,e,e,e,
e,e,w,w,w,w,e,e,
e,w,e,w,w,e,w,e,
w,e,e,w,w,e,e,w,
e,e,e,w,w,e,e,e,
e,e,e,w,w,e,e,e,
e,e,e,w,w,e,e,e,
e,e,e,w,w,e,e,e
]

arrow_red = [ # flecha roja
e,e,e,r,r,e,e,e,
e,e,r,r,r,r,e,e,
e,r,e,r,r,e,r,e,
r,e,e,r,r,e,e,r,
e,e,e,r,r,e,e,e,
e,e,e,r,r,e,e,e,
e,e,e,r,r,e,e,e,
e,e,e,r,r,e,e,e
]
```

```
arrow_green = [ # flecha verde
e,e,e,g,g,e,e,e,
e,e,g,g,g,g,e,e,
e,g,e,g,g,e,g,e,
g,e,e,g,g,e,e,g,
e,e,e,g,g,e,e,e,
e,e,e,g,g,e,e,e,
e,e,e,g,g,e,e,e,
e,e,e,g,g,e,e,e
]

# Fija la variable pause (pausa) en 3 (tiempo inicial entre turnos)

# Fija las variables score (puntaje) y angle (ángulo) en 0

# Crea una variable llamada play en verdadero (True) (esto se usará
para detener el juego)

pause = 3
score = 0
angle = 0
play = True

sense.show_message("Manten la flecha hacia arriba!", scroll_
speed=0.05, text_colour=[100,100,100])

# WHILE play == True

while play:

# ELIGE un angulo al azar

last_angle = angle
while angle == last_angle:
 angle = choice([0, 90, 180, 270])

sense.set_rotation(angle)

# MUESTRA la flecha verde
```

```
sense.set_pixels(arrow)

# DESCANSA por el tiempo de pausa seleccionado

sleep(pause)

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

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

print(angle)
print(x)
print(y)

# SI la orientación es igual a la flecha...

if x == -1 and angle == 180:

# SUMA un punto y pon la flecha verde

sense.set_pixels(arrow_green)
score += 1
elif x == 1 and angle == 0:
sense.set_pixels(arrow_green)
score += 1
elif y == -1 and angle == 90:
sense.set_pixels(arrow_green)
score += 1
elif y == 1 and angle == 270:
sense.set_pixels(arrow_green)
score += 1
else:

# SINO, PON play en `False` y pon la flecha roja

sense.set_pixels(arrow_red)
```

```
play = False

# Acortar el tiempo entre turnos levemente

pause = pause * 0.95

# Esperar antes de la próxima flecha

sleep(0.5)

# Cuando el bucle termine, mostrar un mensaje con el puntaje

msg = "Tu puntaje fue de %s" % score
sense.show_message(msg, scroll_speed=0.05, text_colour=[100, 100,
100])
```

Desafío: más ideas

Ahora que exploraste las posibilidades de tu Sense HAT, quizá quieras investigar otras cosas para hacer:

• Contar un chiste en la pantalla LED.
• Si tu Sense HAT tiene conexión a internet, podrías usar una librería de API de Twitter para mostrar tweets entrantes.
• Creá tus propias imágenes para mostrar en la pantalla LED.
• ¿Podés alternar imágenes para crear una animación?
• Creá un dado electrónico. Agitando el Raspi arroja el dado.
• Creá un termómetro gráfico simple que muestre distintos colores o patrones de acuerdo a la temperatura.
• Escribí un programa que muestre una flecha (o otro símbolo) en la pantalla; este símbolo puede ser usado para indicar la dirección "abajo". De este modo los astronautas en baja gravedad siempre sabrán hacia dónde está la Tierra.
• Usá el acelerómetro para detectar movimientos pequeños - esto puede formar parte de un juego, sistema de alarma, o inclusive de un detector de terremotos.
• Hacé uso del sensor de humedad para detectar aliento, y mostrar un color diferente de acuerdo a la humedad.