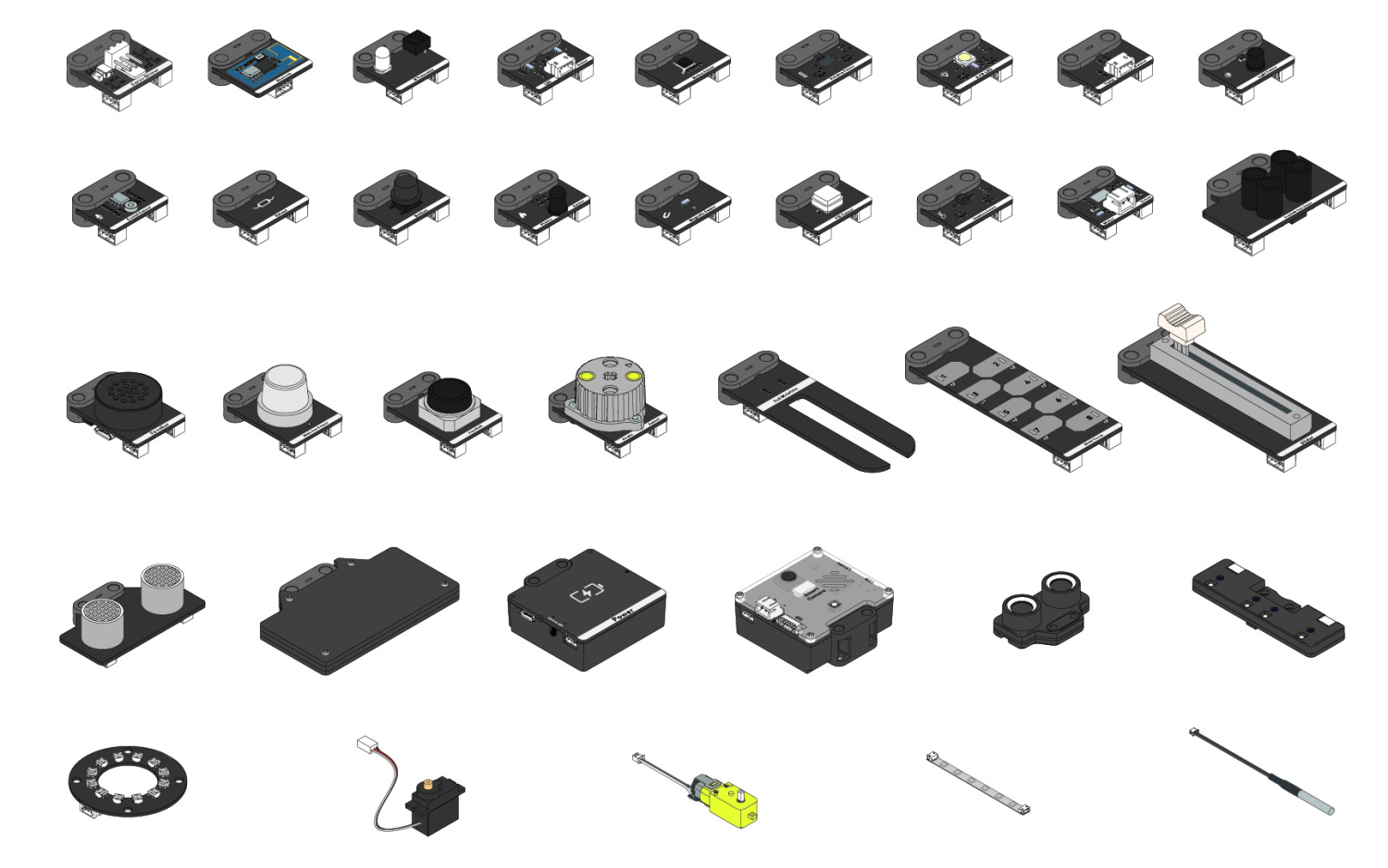
<https://education.makeblock.com/help/mblock-python-editor-apis-for-mbuild-modules>

Los más de 30 módulos mBuild y piezas de apoyo pueden satisfacer sus necesidades de módulos electrónicos en varios escenarios, incluyendo hacer que sus ideas creativas cobren vida, práctica integral, educación basada en proyectos, popularización de programación, popularización de IA y competencias de robótica.  
  


Los módulos mBuild admiten múltiples tableros de control principales y pueden trabajar con ellos a través de diferentes bibliotecas API de Python, con las mismas funciones.  
  
Importe la biblioteca API correspondiente cuando utilice módulos mBuild con una placa de control principal. Además, los módulos mBuild también se pueden utilizar como un dispositivo independiente.  
  
Por ejemplo, para usar módulos mBuild a través de CyberPi, importen la biblioteca *cyberpi.  
  
from cyberpi import \**Para ayudarlo a usar las API de manera más eficiente, los escenarios de aplicación de las API están etiquetados de la siguiente manera:  
  
Py: Indica que una API es compatible con la programación de Python 3 y se puede usar en modo **en vivo** en el editor mBlock Python.  
Mp: Indica que una API es compatible con la programación de MicroPython y se puede usar en el modo de **carga** en el editor mBlock Python.  
Py, Mp: indica que una API es compatible con la programación de Python 3 y MicroPython y se puede usar en los modos **En vivo y Carga** en el editor mBlock Python.

Para usar los módulos mBuild a través de Halocode, importen la biblioteca de *halocode.  
  
de la importación de halocode \**Para usar los módulos mBuild como un dispositivo independiente, importen la biblioteca *mbuild.  
  
From mBuild import \**

Índice de parámetros *index*Todas las API para módulos mBuild incluyen el índice de parámetros, que indica el lugar de un módulo entre los que están conectados a la placa de control principal. Generalmente, el valor predeterminado es 1. Por lo tanto, si solo se usa un módulo de cada tipo, no es necesario configurar este parámetro. Cuando se utilizan dos o más módulos del mismo tipo, deben establecer el índice en 2, 3, 4 u otro número para especificar el segundo, tercero, cuarto u otro módulo. Por ejemplo, *motor\_driver.set(100, index = 2)* indica que la potencia de salida del segundo controlador de motor está establecida en 100.

Módulos de entrada

Módulos de interacción

Botón

Py, Mp: *button.is\_press(index = 1)*Detecta si el botón está presionado.

Devuelve un valor booleano:

*Verdadero*: botón presionado

*Falso:* botón no presionado  
  
Py, Mp: *button.get\_count(index = 1)*  
Obtiene el número de veces que se pulsa el botón después de encenderlo.

Devuelve un valor numérico.  
  
Py, Mp: *button.reset\_count(index = 1)*Restablece el número de veces que se presiona el botón a cero.Sensor de ángulo  
  
Py, Mp:*angle\_sensor.get(index = 1)*  
  
Obtiene el ángulo de rotación del sensor de ángulo (relativo a la posición en la que se encuentra cuando se alimenta).

Devuelve un valor numérico en grados (°). El valor aumenta cuando el sensor de ángulo gira en el sentido de las agujas del reloj y se reduce cuando gira en el sentido contrario a las agujas del reloj.  
  
Py, Mp: *angle\_sensor.reset(index = 1)*Restablece el ángulo que gira el sensor de ángulo.

Py,Mp: *angle\_sensor.get\_speed(index = 1)*

Obtiene la velocidad a la que gira el sensor de ángulo.

Devuelve un valor numérico en grados por segundo (°/s). El valor es positivo cuando el sensor de ángulo gira en sentido horario y negativo cuando gira en sentido antihorario.  
  
Py,Mp: *angle\_sensor.is\_clockwise(index = 1).*  
Detecta si el sensor de ángulo gira en el sentido de las agujas del reloj.

Devuelve un valor booleano:

*Verdadero*: girando en el sentido de las agujas del reloj.

*Falso:* no gira en el sentido de las agujas del reloj.  
  
Py,Mp: *angle\_sensor.is\_anticlockwise(index = 1)*

Detecta si el sensor de ángulo gira en sentido contrario a las agujas del reloj.

Devuelve un valor booleano:

*Verdadero:* girando en sentido contrario a las agujas del reloj

*Falso:* no gira en sentido contrario a las agujas del reloj

Deslizador

Py, Mp: *slider.get(index = 1)*

Obtiene la posición del deslizador.

Devuelve un valor numérico que va de 0 a 100, en porcentaje (%).

Indica la posición del deslizador.  
  
Palanca de mando

Py, Mp: *joystick.get\_x(index = 1)*

Obtiene el valor de salida del joystick en el eje x.

Devuelve un valor numérico que oscila entre –100 y +100.

Py, Mp: *joystick.get\_y(index = 1)*

Obtiene el valor de salida del joystick en el eje y.

Devuelve un valor numérico que oscila entre –100 y +100.

Py,Mp: joystick.is\_up(index = 1)

Determina si el joystick gira hacia arriba.

Devuelve un valor booleano:

*Verdadero:* girando hacia arriba

*Falso:* no gira hacia arriba

Py, Mp: *joystick.is\_down(index = 1)*

Determina si el joystick pivota hacia abajo.

Devuelve un valor booleano:

*Verdadero:* pivotando hacia abajo

*Falso:* no gira hacia abajo

Py, Mp: *joystick.is\_left(index = 1)*

Determina si el joystick pivota hacia la izquierda.

Devuelve un valor booleano:

*Verdadero:* pivotando hacia la izquierda

*Falso:* no pivota hacia la izquierda

Py, Mp: *joystick.is\_right(index = 1)*

Determina si el joystick pivota hacia la derecha.

Devuelve un valor booleano:

*Verdadero:* pivotando hacia la derecha

*Falso:* no pivota hacia la derecha  
  
Multitáctil  
Py,Mp: *multi\_touch.is\_touch(ch = "any", index = 1)*

Detecta si se toca el sensor táctil especificado de Multi Touch. El módulo multi táctil detecta objetos a través del cambio de capacitancia de los sensores táctiles. Por lo tanto, con los ajustes de umbral adecuados, puede detectar el contacto a través de papel, tablas de madera, plásticos u otros aislantes. Después de todo, el acercamiento de un conductor cambia la capacitancia de un sensor táctil.

Parámetros:

*ch:* cadena de caracteres o valor numérico. El valor predeterminado es cualquiera . Para establecer ch en una cadena de caracteres, debe establecerse en any , lo que indica que se puede detectar el toque de cualquier sensor táctil.

Para establecer ch en un valor numérico, se puede establecer en un número del 1 al 8, que indica el número de serie del sensor táctil correspondiente.

Devuelve un valor booleano:

*Verdadero:* tocado

*Falso:* no tocado

Py,Mp: *multi\_touch.reset(level = "middle", index = 1)*

Establece el umbral para Multi Touch.

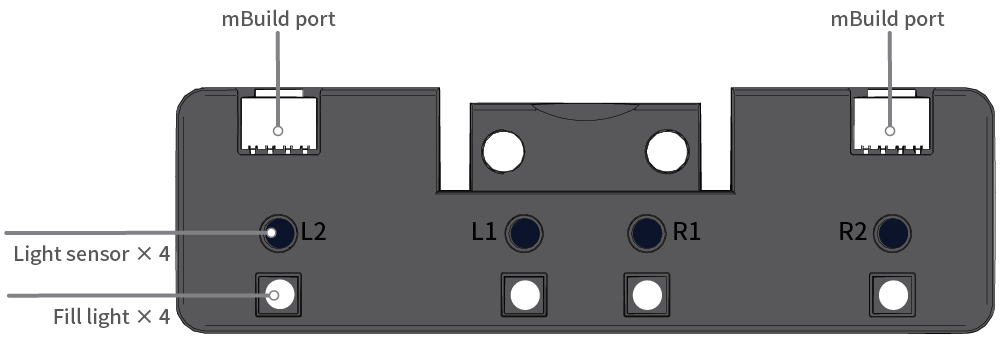
Parámetros:

*nivel:* cadena de caracteres. Los valores se describen de la siguiente manera:

*bajo:* sensibilidad baja

*medio:* sensibilidad moderada

*alto:* sensibilidad alta

Sensor cuádruple RGB  
  
Py,Mp: *quad\_rgb\_sensor.get\_line\_sta(index = 1)*

Obtiene el estado del fondo.

Devuelve un número entero que va de 0 a 15, correspondiente a los números binarios 0000 a 1111, donde 1 indica que se detecta el color de fondo (color claro por defecto) y 0 indica que se detecta la línea (color oscuro por defecto).

La siguiente tabla describe los posibles valores devueltos y los estados de línea correspondientes.

|  |  |  |
| --- | --- | --- |
| **Valor devuelto** | **Número binario** | **Estado de línea detectado**  **(Línea oscura sobre un fondo claro)** |
| 0 | 0000 |  |
| 1 | 0001 |  |
| 2 | 0010 |  |
| 3 | 0011 |  |
| 4 | 0100 |  |
| 5 | 0101 |  |
| 6 | 0110 |  |
| 7 | 0111 |  |
| 8 | 1000 |  |
| 9 | 1001 |  |
| 10 | 1010 |  |
| 11 | 1011 |  |
| 12 | 1100 |  |
| 13 | 1101 |  |
| 14 | 1110 |  |
| 15 | 1111 |  |

Py,Mp: *quad\_rgb\_sensor.get\_offset\_track(index = 1)*

Obtiene cuánto se desvía Quad RGB Sensor de la línea a seguir.

Devuelve un número entero que va de –100 a +100. La desviación de –100 indica que Quad RGB Sensor está a la izquierda de la línea a seguir.

Py,Mp: *quad\_rgb\_sensor.is\_line(ch, index = 1)*

Determina si el sensor especificado detecta una línea.

Parámetro:

*ch:***int o str**, indica el sensor especificado.

Rango de valores: “cualquiera”/”L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor booleano:

*Verdadero:* detectado

*Falso:* no detectado

Py,Mp: *quad\_rgb\_sensor.is\_background(ch, index = 1)*

Determina si el sensor especificado detecta un fondo.

Parámetro:

*ch*: **int o str**, indica el sensor especificado.

Rango de valores: “cualquiera”/”L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor booleano:

*Verdadero*: detectado

*Falso:* no detectado

Py,Mp: *quad\_rgb\_sensor.is\_color(color = "white", ch, index = 1)*

Determina si el sensor especificado detecta el color especificado.

Parámetros:

*color:* cadena, indica el color especificado. A continuación se describen los colores y sus abreviaturas:

r roja

amarillo

g verde

c cian

b azul

p púrpura

blanco

k negro

*ch:* **int o str,** indica el sensor especificado.

*Rango de valores:* “cualquiera”/”L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor booleano:

*Verdadero:* detectado

*Falso:* no detectado

Py,Mp: *quad\_rgb\_sensor.get\_red(ch, index = 1)*

Obtiene el valor R del color detectado por el sensor

Parámetro:

*ch:* **int o str**, indica el sensor especificado.

*Rango de valores*: “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor que va de 0 a 255.

Py, Mp: *quad\_rgb\_sensor.get\_green(ch, index = 1)*

Obtiene el valor G del color detectado por el sensor

Parámetro:

*ch:* **int o str**, indica el sensor especificado.

*Rango de valores:* “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor que va de 0 a 255.

Py,Mp: *quad\_rgb\_sensor.get\_blue(ch, index = 1)*

Obtiene el valor B del color detectado por el sensor.

Parámetro:

*ch:* **int o str**, indica el sensor especificado.

*Rango de valores:* “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor que va de 0 a 255.

Py,Mp: *quad\_rgb\_sensor.get\_gray(ch, index = 1)*

Obtiene la escala de grises detectada por el sensor especificado.

El valor de la escala de grises es la diferencia entre la intensidad de la luz ambiental con la luz de relleno encendida y sin la luz de relleno, lo que indica directamente qué tan bien refleja la luz el objeto.

Parámetro:

*ch:* **int o str**, indica el sensor especificado.

*Rango de valores:* “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor que va de 0 a 100.

Py, Mp: *quad\_rgb\_sensor.get\_color(ch, index = 1)*

Obtiene el valor hexadecimal del color detectado por el sensor especificado.

Parámetro:

*ch:* **int o str,** indica el sensor especificado.

*Rango de valores:* “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor hexadecimal que va de 0x000000 a 0xffffff.

Py,Mp: *quad\_rgb\_sensor.get\_color\_sta(ch, index = 1)*

Obtiene el nombre (cadena de caracteres) del color detectado por el sensor especificado.

Parámetro:

*ch:* **int o str,** indica el sensor especificado.

*Rango de valores:* “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve una cadena de caracteres que puede ser:

"rojo"

"amarillo"

"verde"

"cian"

"azul"

"púrpura"

"blanco"

"negro"

Py,Mp: *quad\_rgb\_sensor.get\_light(ch, index = 1)*

Obtiene la intensidad de la luz ambiental detectada por el sensor especificado.

La luz emitida por las luces de relleno es parte de la luz ambiental. Si es necesario, puede apagar todas las luces de relleno antes de usar esta API.

Parámetro:

*ch:* **int o str**, indica el sensor especificado.

*Rango de valores:* “L2″/”L1″/”R1″/”R2″/”l2″/”l1″/”r1″/”r2″/4/3/2/1

Devuelve un valor que va de 0 a 100.

Py,Mp: *quad\_rgb\_sensor.set\_led(color = "white", index = 1)*

Establece el color de todas las luces de relleno.

Parámetro:

*color*: string o hex, indica el color de las luces de relleno.

Para establecerlo en una cadena, ingrese el nombre completo o la abreviatura de un color; a continuación se describen los colores y sus abreviaturas:

r roja

amarillo

g verde

c cian

b azul

p púrpura

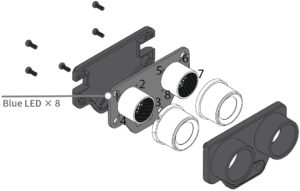
blanco

k negro

Para establecerlo en un valor hexadecimal, ingrese el valor hexadecimal del color.

Py,Mp: *quad\_rgb\_sensor.off\_led(index = 1)*

Apaga las luces de llenado.

Tenga en cuenta que cuando se ejecuta la API para detectar colores, el sensor RGB cuádruple establece automáticamente las luces de relleno en el color que requiere.  
  
Sensor ultrasónico 2  
  
  
Py,Mp: *ultrasonic2.get(index = 1)*

Obtiene la distancia entre un obstáculo y el sensor ultrasónico 2.

Devuelve un valor entre 3 y 300, en centímetros, con un error de ±5%.

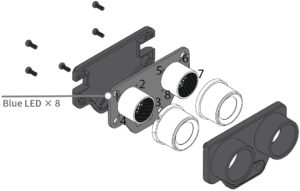
Py,Mp: *ultrasonic2.led\_show(bri, index =1)*

Establece el brillo de todos los LED.

Parámetro:

*bri:* int\_list

Este parámetro debe configurarse en el formato de [ bri1,bri2,bri3,bir4,bri5,bri6,bri7,bri8], donde x indica el número de un LED.

La siguiente figura muestra los LED y sus números.  
  
Py,Mp: *ultrasonic2.set\_bri(led\_bri, id, index =1)*

Establece el brillo del LED especificado.

Parámetros:

*led\_bri float:* brillo del LED especificado, que va de 0 a 100

*id:* int o str

*int:* número del LED especificado, que va de 1 a 8

*str:* El valor solo puede ser all, indicando todos los LEDs.

Py,Mp: *ultrasonic2.add\_bri(led\_bri, id, index =1)*

Aumenta el brillo del LED especificado.

Parámetros:

*led\_bri float:* valor por el que se quiere aumentar la luminosidad, que va de 0 a 100.

*id:* int o str

*int:* número del LED especificado, que va de 1 a 8

*str:* El valor solo puede ser all, indicando todos los LEDs.

Py,Mp: *ultrasonic2.get\_bri(id, index =1)*

Obtiene el brillo del LED especificado .

Parámetro:

*id:* int o str

*int:* número del LED especificado, que va de 1 a 8

*str:* El valor solo puede ser all, indicando todos los LEDs.

Sensor de luz

Py,Mp: *light\_sensor.get(index = 1)*

Obtiene el valor de salida del sensor de luz.

Devuelve un valor numérico que va de 0 a 100.  
  
sensor de sonido

*sound\_sensor.get(index = 1)*

Obtiene el valor de salida del sensor de sonido.

Devuelve un valor numérico que va de 0 a 100.  
  
Sensor ultrasónico

Py,Mp: *ultrasonic.get(index = 1)*

Obtiene la distancia entre el sensor ultrasónico y el obstáculo.

Devuelve un valor numérico que va de 3 a 300, en centímetros (cm), con una desviación de ±5%.

Sensor de alcance

Py,Mp: *ranging\_sensor.get(index = 1)*

Obtiene la distancia entre el sensor de distancia y el obstáculo.

Devuelve un valor numérico que va de 2 a 200, en centímetros (cm), con una desviación de ±5%.  
  
Sensor de movimiento

Py,Mp: *motion\_sensor.is\_shake(index = 1)*

Detecta si el sensor de movimiento se sacude.

Devuelve un valor booleano:

cierto: sacudido

Falso: no agitado

Py,Mp: *motion\_sensor.get\_shakeval(index = 1)*

Obtiene la fuerza con la que se agita el sensor de movimiento.

Devuelve un valor numérico que oscila entre 0 y 100. Un valor mayor indica una sacudida más fuerte.

Py,Mp: *motion\_sensor.get\_accel(axis, index = 1)*

Obtiene la aceleración en el eje x, y o z.

Parámetros:

*eje*: cadena de caracteres, que puede ser x, y o z, que indica un eje definido para el sensor de movimiento.

Devuelve un valor numérico en metros por segundo al cuadrado (m/s²).

Py,Mp: *motion\_sensor.is\_tiltleft(index = 1)*

Detecta si el sensor de movimiento está inclinado hacia la izquierda. El umbral es de 15 grados.

Devuelve un valor booleano:

*Verdadero*: inclinado hacia la izquierda

*Falso:* no inclinado hacia la izquierda

Py,Mp: *motion\_sensor.is\_tiltright(index = 1)*

Detecta si el sensor de movimiento está inclinado hacia la derecha. El umbral es de 15 grados.

Devuelve un valor booleano:

*Verdadero:* inclinado hacia la derecha

*Falso:* no inclinado hacia la derecha

Pu,Mp: *motion\_sensor.is\_tiltup(index = 1)*

Detecta si el sensor de movimiento está inclinado hacia arriba. El umbral es de 15 grados.

Devuelve un valor booleano:

*Verdadero:* inclinado hacia arriba

*Falso:* no inclinado hacia arriba

Py,Mp: *motion\_sensor.is\_tiltdown(index = 1)*

Detecta si el sensor de movimiento está inclinado hacia abajo. El umbral es de 15 grados.

Devuelve un valor booleano:

*Verdadero:* inclinado hacia abajo

*Falso:* no inclinado hacia abajo

Py,Mp: *motion\_sensor.is\_faceup(index = 1)*

Detecta si el sensor de movimiento está colocado boca arriba.

*Verdadero:* colocado boca arriba

*Falso:* no se coloca boca arriba

Py,Mp: *motion\_sensor.is\_facedown(index = 1)*

Detecta si el sensor de movimiento está colocado boca abajo.

*Verdadero:* colocado boca abajo

*Falso:* no se coloca boca abajo

Py,Mp: *motion\_sensor.get\_roll(index = 1)*

Obtiene el ángulo de balanceo del sensor de movimiento.

Devuelve un valor numérico que oscila entre –90 y +90, en grados (°).

Py,Mp: *motion\_sensor.get\_pitch(index = 1)*

Obtiene el ángulo de cabeceo del sensor de movimiento.

Devuelve un valor numérico que oscila entre –180 y +180, en grados (°).

Py,Mp: *motion\_sensor.get\_yaw(index = 1)*

Obtiene el ángulo de guiñada del sensor de movimiento.

Devuelve un valor numérico que va de 0 a 360, en grados (°).

**Nota:** No se utiliza una brújula electrónica y, por lo tanto, el ángulo de guiñada es la integral de la velocidad angular del sensor de movimiento en el eje z. Puede ocurrir un error acumulativo. Esta función API no se puede utilizar para obtener el ángulo de guiñada preciso.

Py,Mp: *motion\_sensor.get\_gyro(axis, index = 1)*

Obtiene la velocidad angular del sensor de movimiento en el eje x, y o z.

Parámetros:

*eje:* cadena de caracteres, que puede ser x, y o z, que indica un eje definido para el sensor de movimiento.

Devuelve un valor numérico en grados por segundo (°/s).

Py,Mp: *motion\_sensor.get\_rotation(axis, index = 1)*

Obtiene el ángulo de rotación del sensor de movimiento en el eje x, y o z.

El ángulo es positivo cuando el sensor de movimiento gira en sentido contrario a las agujas del reloj.

Parámetros:

*eje:* cadena de caracteres, que puede ser x, y o z, que indica un eje definido para el sensor de movimiento.

Devuelve un valor numérico en grados (°).

Py,Mp: *motion\_sensor.reset\_rotation(axis= "all", index = 1)*

Inicializa el ángulo actual del sensor de movimiento en los ejes x, y y/o z como cero. Después de la inicialización,

*get\_rotation()* calcula desde cero.

Parámetros:

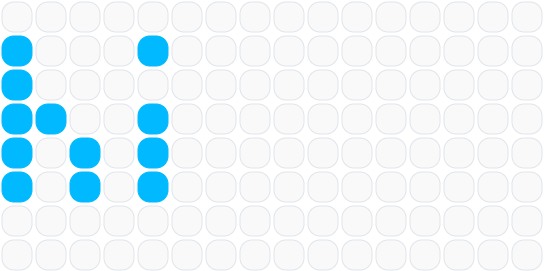
*eje:* cadena de caracteres, que puede ser x, y, z o todos. x, y y z son los ejes definidos para el sensor de movimiento. El valor predeterminado es all, que indica los tres ejes.  
  
Módulos de salida  
Luz

Matriz LED  
  
Py,Mp: *led\_matrix.show(image = "hi", index = 1)*

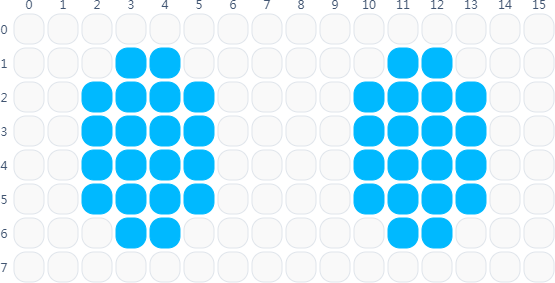
Establece la imagen que se mostrará en la matriz de LED.

Parámetros:

imagen: cadena de caracteres. El valor predeterminado es hola . La siguiente figura muestra la imagen en la matriz de LED cuando la imagen está configurada en hi.



Para mostrar una imagen que no está predefinida, debe configurar la imagen en el número hexadecimal correspondiente a la imagen. Los números hexadecimales se obtienen de la siguiente manera:

Tome el estado de encendido/apagado de los ocho LED en una columna como un número binario de 8 dígitos. Convierta el número binario de 8 dígitos en un número hexadecimal de 2 dígitos. Hay 16 columnas y, por lo tanto, se puede usar un número hexadecimal de 32 dígitos para indicar el estado de encendido/apagado de los LED. La siguiente figura muestra los LED en la matriz de LED.  
  
Programa de ejemplo 1:

*from halocode import \**

*led\_matrix.show(image = "hello", index = 1) #Enables the LED matrix to display the image "hello"*

Programa de ejemplo 2:

*from halocode import \**

*led\_matrix.show(image = "00003c7e7e3c000000003c7e7e3c0000", index = 1)*

*#Permite que la matriz LED muestre la imagen "hola"*Py,Mp: *led\_matrix.print(message, index = 1)*

Muestra los caracteres especificados en la matriz de LED en modo de desplazamiento.

Parámetros:

mensaje: cadena de caracteres. El valor puede ser una combinación de cualquier carácter, dígito y puntuación en inglés. Los caracteres chinos no son compatibles. Cuando se excede el rango de visualización de la matriz LED, se muestra un mensaje en modo de desplazamiento.Py,Mp: *led\_matrix.print\_until\_done(message, index = 1)*

Muestra los caracteres especificados en la matriz de LED hasta que finaliza el desplazamiento.

Parámetros:

mensaje: cadena de caracteres. El valor puede ser una combinación de cualquier carácter, dígito y puntuación en inglés. Los caracteres chinos no son compatibles. Cuando se excede el rango de visualización de la matriz LED, se muestra un mensaje en modo de desplazamiento. El hilo está bloqueado hasta que finaliza el desplazamiento.  
  
Py,Mp: *led\_matrix.print\_at(message, x, y, index = 1)*

Muestra los caracteres especificados con el punto de inicio especificado (x,y) en la matriz de LED.

Parámetros:

*mensaje:* cadena de caracteres. El valor puede ser una combinación de cualquier carácter, dígito y puntuación en inglés. Los caracteres chinos no son compatibles. Cuando se excede el rango de visualización de la matriz LED, se muestra un mensaje en modo de desplazamiento. El hilo se bloquea hasta que finaliza el desplazamiento.

x: valor numérico que va de 0 a 15, que indica la coordenada x inicial

y: valor numérico que va de 0 a 7, que indica la coordenada y de inicio

Py,Mp: *led\_matrix.on(x, y, index = 1)*

Enciende el LED en la posición especificada (x,y) en la matriz de LED.

Parámetros:

*x:* valor numérico de 0 a 15, que indica la coordenada x del led a encender

*y:* valor numérico de 0 a 7, que indica la coordenada y del LED a encender  
  
Py,Mp: *led\_matrix.off(x, y, index = 1)*

Apaga el LED en la posición especificada (x,y) en la matriz de LED.

Parámetros:

*x:* valor numérico de 0 a 15, que indica la coordenada x del LED a apagar

*y:* valor numérico de 0 a 7, que indica la coordenada y del LED a apagar  
  
Py,Mp: *led\_matrix.toggle(x, y, index = 1)*

Cambia el estado (encendido/apagado) del LED en la posición especificada (x,y) en la matriz de LED.

Parámetros:

*x:* valor numérico de 0 a 15 que indica la coordenada x del LED cuyo estado se desea cambiar

*y:* valor numérico de 0 a 7 que indica la coordenada y del LED cuyo estado se desea cambiar

Py,Mp: *led\_matrix.clear(index = 1)*

Apaga todos los LED de la matriz de LED.  
  
LED RGB