

Lección 4:

Monitorización medioambiental

Programación basada en bloques

Drones4STEM © 2023 por Drones4STEM tiene licencia CC BY-NC-SA 4.0. Para ver una copia de esta licencia, visite <https://creativecommons.org/licenses/by-nc-sa/4.0/>

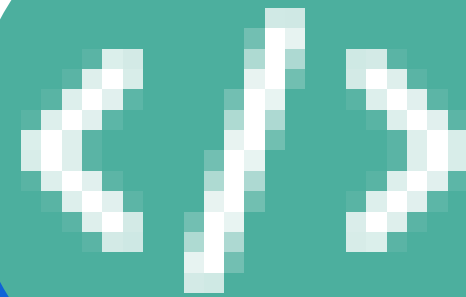


Co-funded by
the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Objetivos de la lección

Aprende los conceptos básicos de la programación basada en bloques



SCRATCH

¡Aprende a usar Scratch y Mind+ para programar la misión!

¡Ponte a prueba programando una misión con drones!



Introducción a la programación basada en bloques

¿Qué es la programación basada en bloques?

Definición: método de codificación visual para programar sin sintaxis, conectando instrucciones en forma de bloques.

Objetivo principal: Ser un enfoque de programación fácil para principiantes, centrándose en los conceptos básicos de estructura, lógica y resolución de problemas.



¿Por qué utilizar la programación basada en bloques?

1 Simplifica la programación.

- Permite a los principiantes escribir programas perfectamente completos sin preocuparse por los detalles de la sintaxis y otras complejidades de la programación basada en texto.

2 Aprendizaje visual

- Permite a los programadores de cualquier edad comprender visualmente la estructura y las conexiones de la programación a través de la representación en bloques.

3 Fomenta la experimentación y la creatividad

- Con una herramienta tan fácil de usar, los programadores sin experiencia pueden probar rápidamente nuevas ideas y poner a prueba sus habilidades para resolver problemas.

Herramientas para la programación basada en bloques

1 Scratch

- Un popular lenguaje de programación basado en bloques con su propio entorno de desarrollo. ¡De uso gratuito, apto para principiantes y con muchas aplicaciones prediseñadas que se pueden encontrar en el sitio web oficial!

2 Mind+

- Mind+ es un software de programación gratuito basado en Scratch 3.0 que admite diversos tipos de hardware de código abierto, como Arduino, micro:bit, mPython-esp32 y todo tipo de placas controladoras.
- Mind + es compatible con docenas de bibliotecas de módulos de extensión de uso común, lo que lo hace adecuado para la programación de drones.

Tipos de bloques

1

MOTION

It allows you to position and move the sprite within the work area.

2

LOOKS

It allows you to change the appearance and let the sprite speak.

3

SOUND

It allows you to introduce sound effects.

4

EVENTS

It allows to start the program according to the different events and manages the messages.

5

CONTROL

It allows you to make loops and insert conditional blocks.

6

SENSING

It allows you to manage interactions.

7

OPERATORS

It allows you to perform mathematical and comparison operations.

8

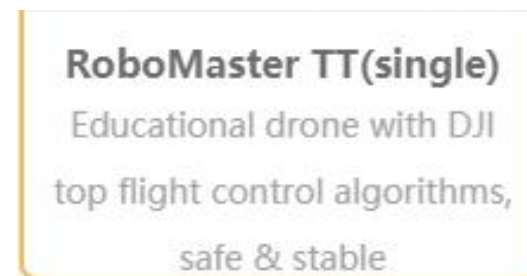
VARIABLES

It allows you to create and manage variables.

Programación basada en bloques y drones

La programación basada en bloques se puede utilizar para programar dispositivos drones, indicándoles que sigan diferentes rutas de vuelo, realicen despegues y aterrizajes, utilicen sensores, tomen fotografías y mucho más.

Para programar el dron Robomaster TT en Mind+, primero debe instalar la extensión correcta:



Seleccione

luego

luego

En Mind+, los bloques utilizados para programar drones son

y se pueden encontrar



sección después de instalar la extensión.

Ejemplo sencillo de programación de drones basada en bloques

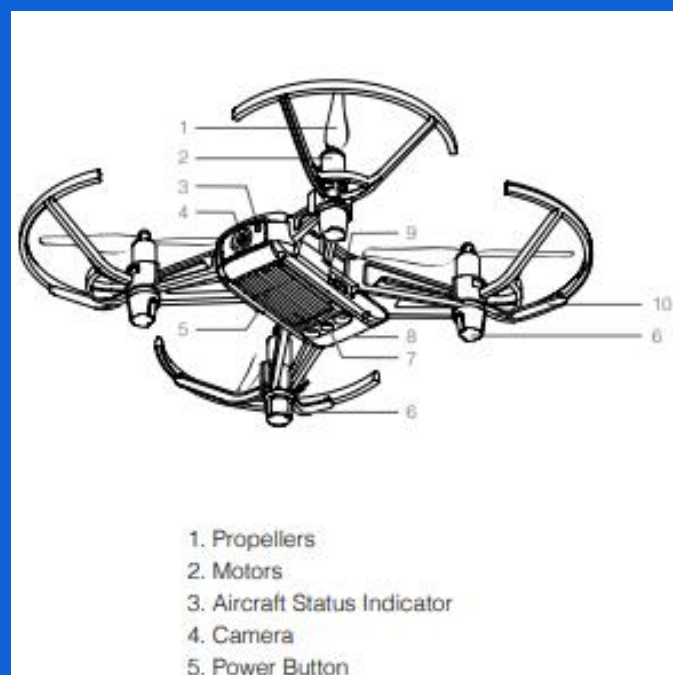
Los bloques se conectan entre sí como un rompecabezas.

El programador puede arrastrar y soltar bloques para crear su programa.

Este sencillo ejemplo muestra lo fácil que es utilizar bloques para programar un dron para que despegue, vuele 50 cm hacia la izquierda, vuele 50 cm hacia la derecha, repita eso 10 veces y luego aterrice.



Caso de uso 2 Misión: Conectar el dron



Paso 1

Pulse el botón de encendido del dron.

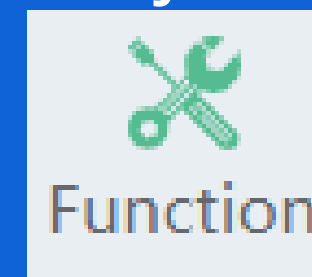
Conéctese al dron a través de wifi, a la red TELLO o RMTT.



Paso 2

Paso 3

Vaya a



en Mind+ y pulse

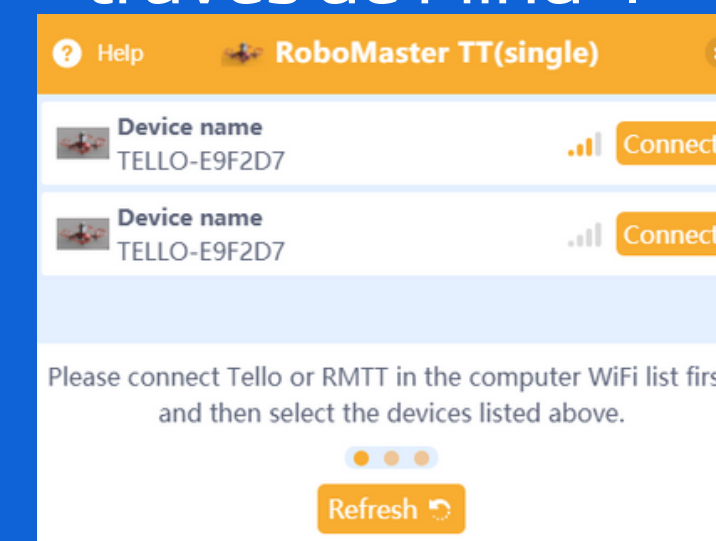
el



▶ RoboMaster TT(single)



Conéctese al dispositivo a través de Mind+.



Paso 4

Caso práctico 2 Misión: Programación de drones y sensores

Objetivo:

El objetivo de la misión es:

- Enseñar la programación de drones basada en bloques
- Enseñar el uso de sensores con drones
- Enseñar la recopilación y el análisis

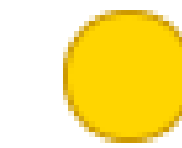
En esta lección de programación:

- Programar un dron Robomaster TT utilizando Mind+
- Recopilar datos de los sensores internos del dron
- Organizar los datos en listas para facilitar su análisis

Caso de uso 2 Misión: Conceptos básicos sobre el vuelo con drones

Para empezar

Utilizando comandos sencillos que se encuentran en el,



Events

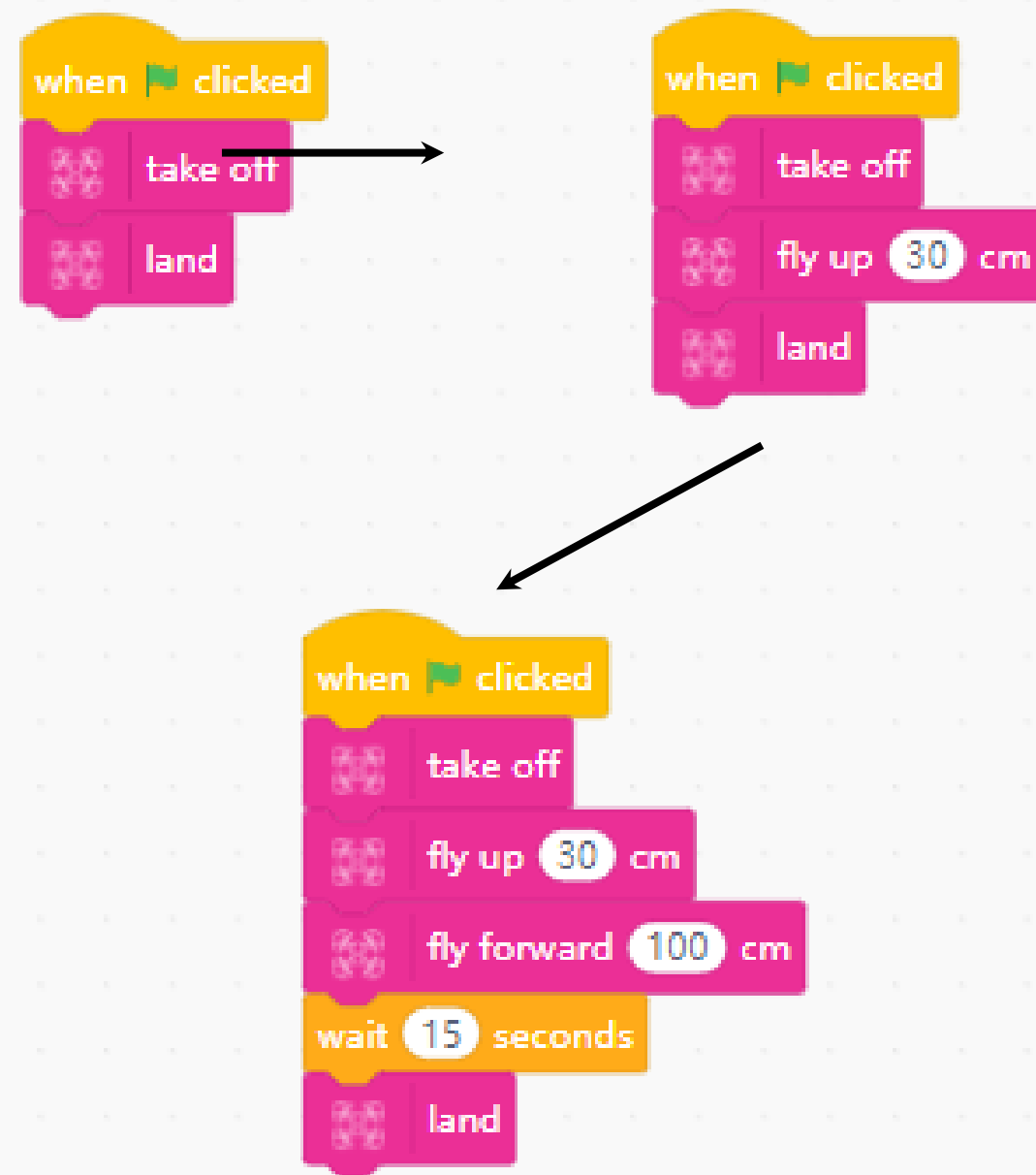


Function

creamos poco a poco un programa sencillo para que el dron:

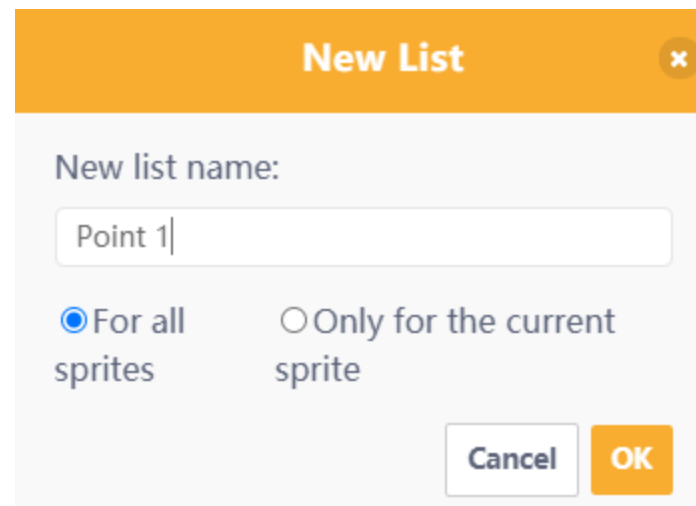
- Despegar y volar 30 cm hacia arriba
- Volar 100 cm hacia delante y esperar 15 segundos
- Aterrizar

(1)

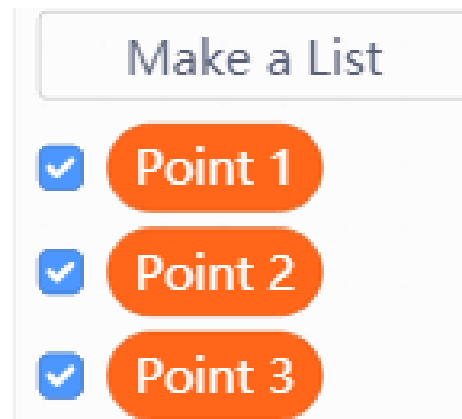


Caso de uso 2 Misión: Crear listas

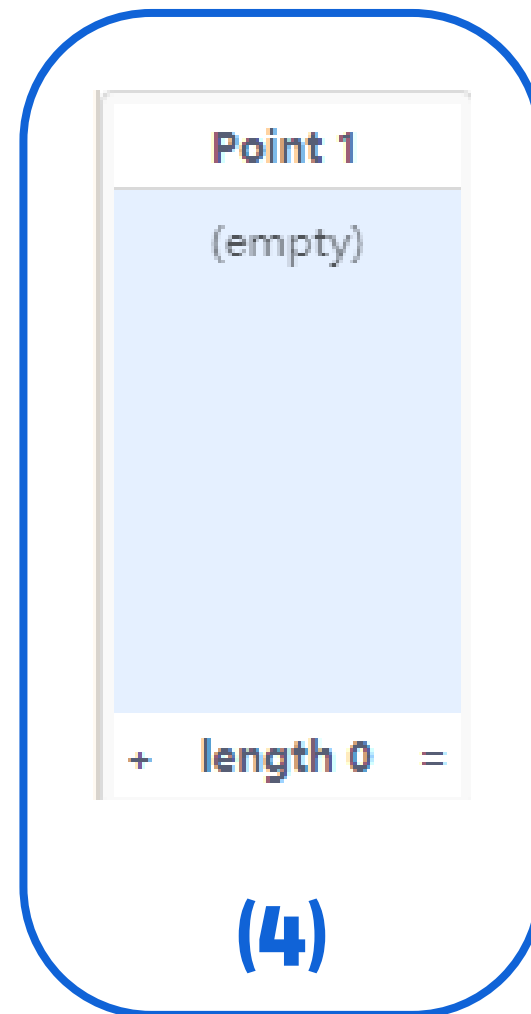
(2)



(3)



(4)



Creación de listas

Antes de usar Listas para almacenar datos, tenemos que crearlas. En la sección

Make a List



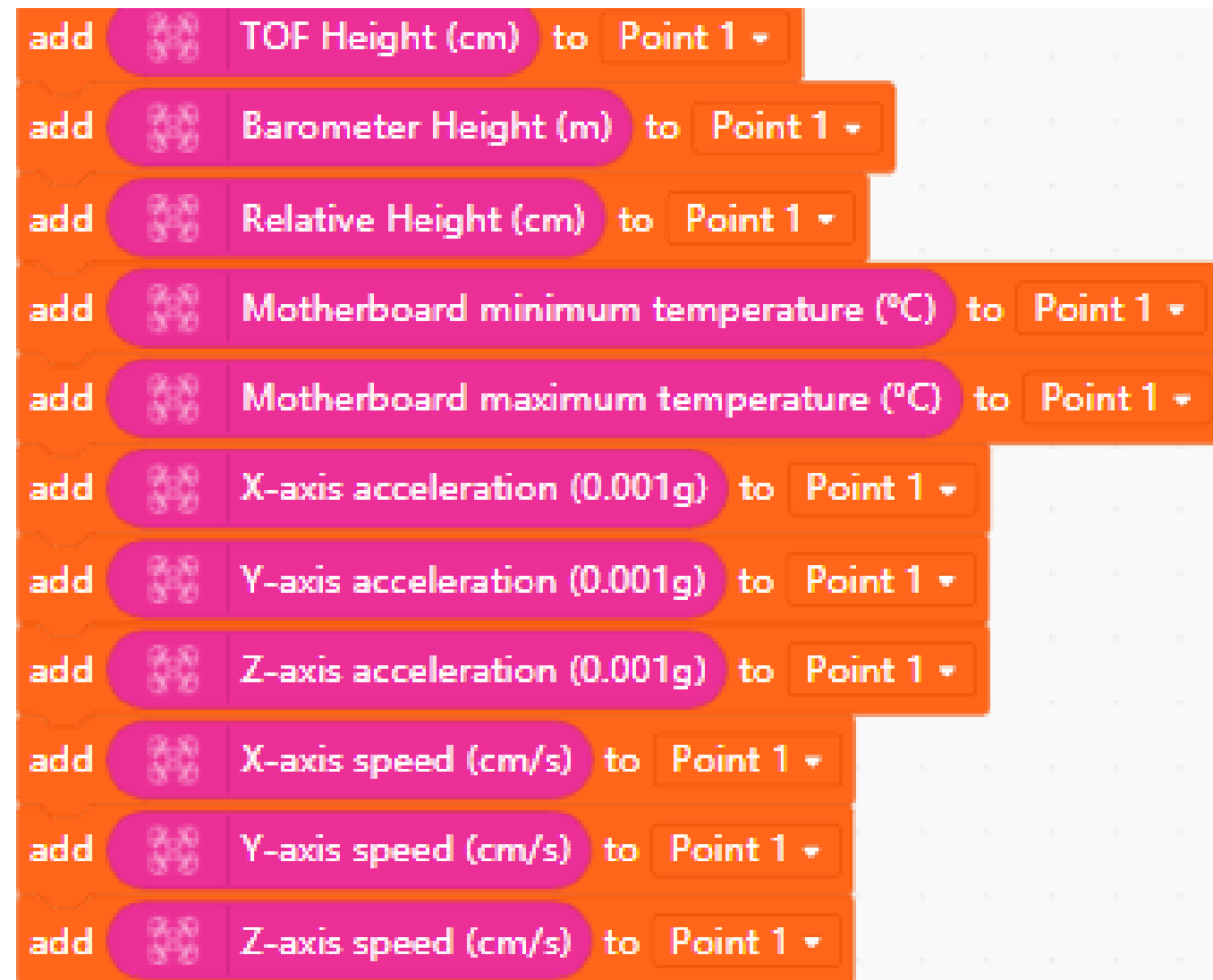
encontramos el comando que provoca la ventana emergente (2).

Usamos este comando para crear tres listas: Punto 1, Punto 2, Punto 3, como se ve en (3).

Estas listas aparecen vacías en la sección de sprites, como se ve en (4).

Caso de uso 2 Misión: Rellenar listas con datos

(5)

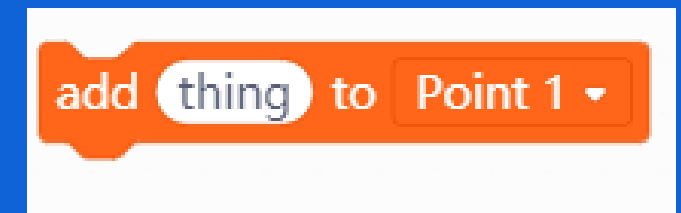


Datos en listas

Después de crear las listas, en la sección



encontramos el comando



que podemos usar para poner diferentes variables en las listas.

En la sección

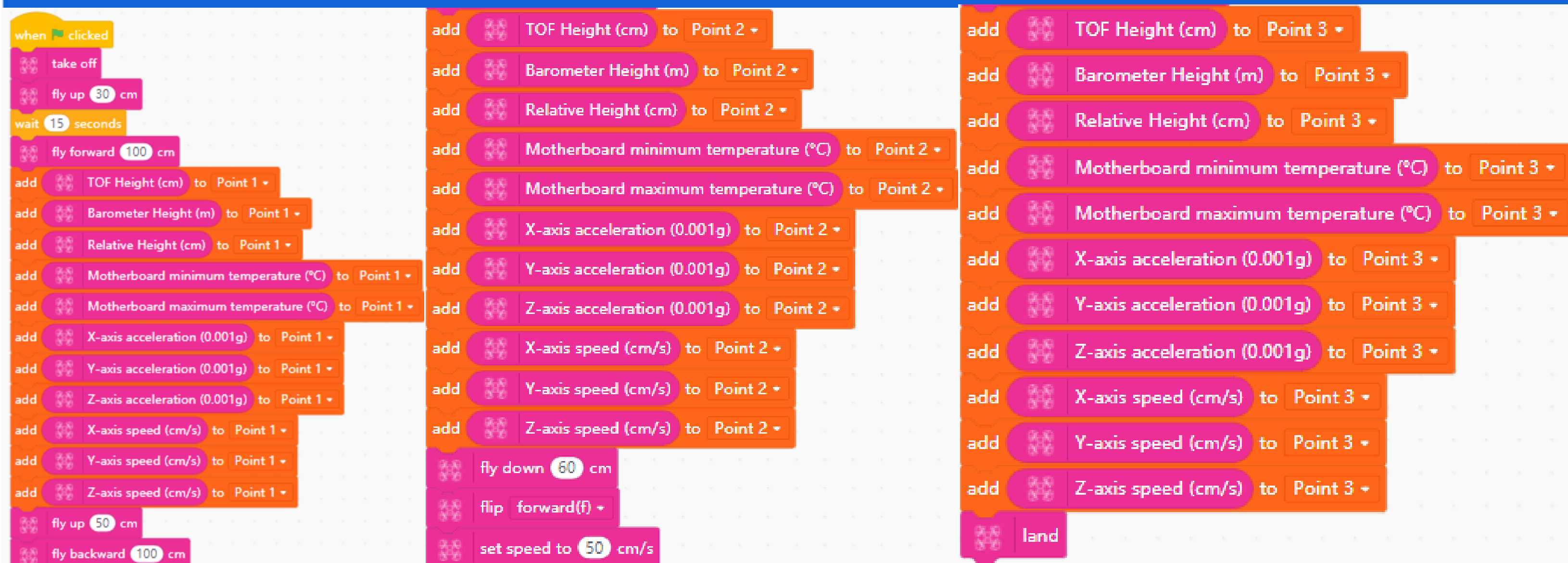


, encontramos variables

como



Caso de uso 2 Misión: Programa final



Combinación

Repetimos el proceso del punto (5) para las listas del punto 2 y del punto 3. A continuación, combinamos esos comandos con nuestro programa sencillo del punto (1).

Entre las mediciones, añadimos diferentes acciones del dron para garantizar que haya variedad en los datos

Caso de uso 2 Misión: Resultado final

Fin del programa

Después de ejecutar el programa, tenemos tres listas diferentes con mediciones procedentes de los sensores del dron.

¡Utilizaremos estos datos en las siguientes etapas de la misión!

El proceso se puede replicar utilizando sensores externos en lugar de internos, para recopilar datos relacionados con el entorno.

Point 1		Point 2		Point 3	
1	107	1	220	1	154
2	37.84	2	38.22	2	38.02
3	80	3	130	3	120
4	71	4	71	4	69
5	74	5	73	5	72
6	-32.00	6	-32.00	6	-34.00
+ length 11 =		+ length 11 =		+ length 11 =	

Caso de uso 2 Misión: Bonificación

Uso de sensores externos

Para utilizar sensores externos, añadiremos una sección de bonificación al caso de uso. Para ello se necesita:

- Un sensor de temperatura y humedad DHT11
- Una placa de desarrollo ESP32
- Una placa de conexión para tarjeta micro SD mini
- Una batería (Li-Po - 3,7 V, 800 mAh)
- Un cargador de batería (1 A USB-C - TP4056)
- Un interruptor (Mini SPDT - 1 A/250 VCA)



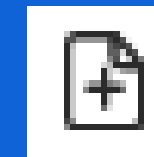
Caso de uso 2 Misión: Bonificación

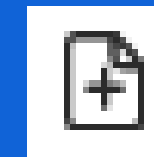
Programación



Para utilizar un sensor externo, nos trasladaremos a la sección Código de Mind+ con el fin de escribir código Python.

El archivo `main.py` del caso de uso es el que utilizaremos.



Para crear un nuevo archivo Python, haga clic en el icono  junto a «Archivo en el proyecto» y, a continuación, escriba o pegue su código.

El archivo `main.py` programará la recopilación de datos del sensor.

Caso de uso 2 Misión: Bonificación

```
import machine
import dht
import time
import os
from machine import SPI, Pin
import sdcard

# DHT22 Sensor Setup
DHT_PIN = 4
dht_sensor = dht.DHT22(machine.Pin(DHT_PIN))

# SD Card Configuration
CS_PIN = 5 # Confirm this matches your wiring
spi = SPI(1,
          baudrate=400000, # Reduced baudrate for stability
          sck=Pin(13),
          mosi=Pin(15),
          miso=Pin(14))

try:
    sd = sdcard.SDCard(spi, Pin(CS_PIN))
    os.mount(sd, '/sd')
    print("SD card mounted successfully")
    print("Root directory:", os.listdir('/'))
    print("SD card contents:", os.listdir('/sd'))
except Exception as e:
    print("SD card initialization failed:", e)
    machine.reset()
```

```
# File Configuration
FILE_PATH = '/sd/data.csv'

# Test file operations
try:
    with open(FILE_PATH, 'a') as f:
        f.write('Test entry\n')
        print("File write test successful")
except Exception as e:
    print("File operation failed:", e)

def log_data(temp, hum):
    try:
        timestamp = "{:02d}::{:02d}::{:02d}".format(*time.localtime())
        data = f"Temp {temp:.1f},Hum {hum:.1f}\n"
        with open(FILE_PATH, 'a') as f:
            f.write(data)
        print(f"Data logged: {data.strip()}")
    except Exception as e:
        print("Logging failed:", e)

while True:
    try:
        dht_sensor.measure()
        temp = dht_sensor.temperature()
        hum = dht_sensor.humidity()
        log_data(temp, hum)
    except OSError as e:
        print("Sensor read error:", e)

    time.sleep(10)
```

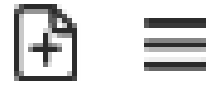
El código

El código Python en main.py se encarga de lo siguiente:

- Configuración del sensor DHT11
- Configurar la tarjeta SD
- Crear un archivo .csv para los datos
- Registrar las mediciones del sensor en el archivo .csv

Caso de uso 2 Misión: Bonificación

File in the project



 main.py

 sdcard.py

Biblioteca SD

Para que el código de main.py funcione, se necesita una biblioteca adicional.

La biblioteca sdcard.py se puede encontrar en este enlace: <https://github.com/micropython/micropython-lib/blob/master/micropython/drivers/storage/sdcard/sdcard.py>

Copie el código y péguelo en un nuevo archivo Python llamado sdcard.py

Caso de uso 2 Misión: Bonificación

Run Tools Help

Configure interpreter...

Which kind of interpreter should Thonny use for running your code?

MicroPython (ESP32)

File Edit View Run Tools Help

New	Ctrl+N
Open...	Ctrl+O
Recent files	
Close	Ctrl+W
Close all	Ctrl+Shift+W
Save	Ctrl+S
Save All files	Ctrl+Alt+S
Save as...	Ctrl+Shift+S



Cargar código en la placa de desarrollo

Una forma de guardar nuestros archivos main.py y sdcard.py en la placa de desarrollo es a través del IDE Thonny, siguiendo estos pasos:

- Paso 1: Abra el archivo main.py con Thonny.
- En Thonny, haz clic en Ejecutar, luego en Configurar intérprete y selecciona MicroPython ESP32. Pulsa Aceptar.
- A continuación, haz clic en Archivo, Guardar como y selecciona la opción MicroPython.
- Repita el proceso con el archivo sdcard.py.

Ahora, el código está en la placa de desarrollo y se ejecutará automáticamente cada vez que encienda el interruptor.

Preguntas

- ¿Cómo hace la programación basada en bloques que la programación sea más accesible?
- ¿Por qué el dron tiene sensores internos?
- ¿Cómo se puede utilizar la programación basada en bloques de drones en la monitorización medioambiental?

¿Qué preguntas tienes?



Recursos adicionales

Lecturas recomendadas

<https://mindplus.dfrobot.com/RMTT>

Vídeos

[Presentación de Mind+: un software de programación Sctrach 3.0](#)



¡Gracias por su atención!



Co-funded by
the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

