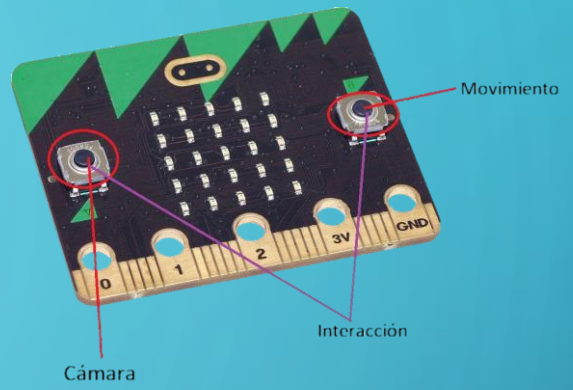
MICRO:BIT


USB
BLE ANTENNA
RESET
BATTERY
BBC
PROCESSOR
COMPASS
ACCELEROMETER
PINS
micro:bit

INTRODUCCIÓN Y OBJETIVOS



- “Micro:bit es una pequeña computadora programable, diseñada para hacer que el aprendizaje y la enseñanza sean fáciles y divertidos”.
- Gracias a la gran cantidad de sensores que incorpora, sólo con la placa se pueden llevar a cabo centenares de proyectos. BBC micro:bit también es una plataforma IoT (Internet of Things), lo que la hace muy interesante para usuarios avanzados. Tanto el hardware como el software de micro:bit es de código abierto.
- En este tercer taller se mostrará como convertir el micro:bit en un controlador para una experiencia de realidad virtual creada con A-Frame. La continuidad de este taller con el anterior se manifiesta en la implementación el micro:bit como controlador en la experiencia, pero además el micro:bit debe funcionar en cualquier entorno de A-Frame que permita movimiento, independientemente de como sea la experiencia.
- Las funciones que llevará cabo el micro:bit al acabar el taller, serán las siguientes:
    1. Movimiento físico en la experiencia al pulsar el botón derecho **B**.
    2. Movimiento de la cámara al pulsar el botón izquierdo **A**.
    3. Interacción al pulsar los dos botones a la vez del micro:bit.

PASOS PREVIOS

- Para la realización de este taller es necesario tener algunos conocimientos básicos sobre Python.
- Para programar el micro:bit primero se debe preparar el entorno, el lenguaje utilizado será una variante de Python propia del micro:bit que se conoce como micropython. El sistema operativo será Linux, aunque micro:bit también dispone de compatibilidad con Windows mediante la instalación de un driver.
- Los requisitos antes de empezar a programar el micro:bit son:
  - Disponer de un editor de texto.
  - Instalar bitio.
  - Instalar pyautogui.

INSTALACIÓN DE BITIO

1. El primer paso es instalar Bitio, para ello lo primero que se debe hacer es ir a su repositorio de GitHub.
2. El siguiente paso es copiar la carpeta 'src/microbit' en el directorio Home del proyecto actual. Con esto se consigue que automáticamente cualquier programa que este en el mismo directorio acceda a la carpeta de bitio cuando se importe el paquete microbit, al principio del programa.

   import microbit

3. Si bitio está correctamente instalado la salida al ejecutar un programa que lo use debería ser parecida a esta.

   No micro:bit has previously been detected

   Scanning for serial ports

   remove device, then press ENTER

   scanning...

4. Para ejecutar todos los programas es necesario tener el micro:bit desconectado antes de lanzarlo, a continuación la terminal pedirá que se conecte, para así reconocerlo y correr el programa posteriormente.

✓ Ejercicio1

- Descarga algún programa de la carpeta /src y prueba las diferentes funcionalidades que ofrecen, se ejecutan como cualquier programa de python: $ **python programa.py**

INSTALACIÓN DE PYAUTOGUI

- A continuación se debe instalar pyautogui para poder mapear las funciones del teclado y el ratón al micro:bit.
- Estos son los comandos necesarios para instalarlo:

```
$ pip3 install python3-xlib
$ sudo apt-get install scrot
$ sudo apt-get install python3-tk
$ sudo apt-get install python3-dev
$ pip3 install pyautogui
```

- Para usar pyautogui basta con importarlo al principio del programa, de la misma forma que se ha mostrado con bitio.

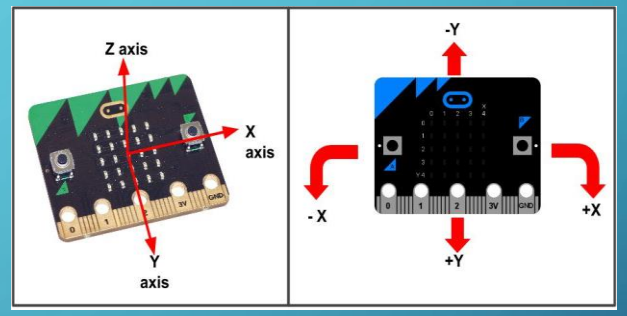
```
import pyautogui
```

✓ Ejercicio2

❖ Descarga el programa que se adjunta con el taller testpy.py y ejecútalo para comprobar que se ha instalado pyautogui correctamente. Una vez esté el programa en ejecución el ratón debe moverse al centro de la pantalla al pulsar el botón **A** del micro:bit.

PROGRAMANDO EL MICRO:BIT

- Una vez se tiene el entorno preparado y todas las herramientas instaladas, se puede proceder a programar el micro:bit y convertirlo en el controlador de la experiencia.
- Las funciones a implementar son:
  A. Movimiento
  B. Interacción.
  C. Control de la cámara.



A. Movimiento

- Para moverse en A-Frame, se utilizan las flechas de dirección. El objetivo por tanto, es mapear los valores del acelerómetro a una de las teclas de dirección, es decir, si el micro:bit se mueve a izquierda o derecha se "pulsarán", gracias a pyautogui, las teclas izquierda o derecha del teclado.
- Cuando se ejecuta la función que nos proporciona los valores del acelerómetro, por ejemplo en el eje X microbit.accelerometer.get_x() , se observa cuando se mueve a la derecha que los valores van de 0 a 1000 en la parte positiva y de -1000 a 0 en la parte negativa, cuando se mueve a la izquierda. Por tanto cuando se mueve hacia arriba en el eje Y se reciben valores positivos y moviéndolo hacia abajo, valores negativos.
- Se establece un valor de Threshold de 200, con el objetivo de prevenir las pequeñas desviaciones no deseadas en alguno de los ejes al mover el micro:bit, de tal manera que al recibir valores mayores a 200 en alguno de los ejes, la tecla correspondiente es pulsada mediante la función pyautogui.keyDown( ).

MOVIMIENTO

- Este es el programa que implementa la funcionalidad del movimiento de la cámara:

```
def Move():
    while True:
        x = microbit.accelerometer.get_x()
        %% = microbit.accelerometer.get_%%()

        if x > %%:
            pyautogui.keyDown(%%)
        elif x < -THRESHOLD:
            pyautogui.keyDown(%%)
        else:
            pyautogui.keyUp("right")
            pyautogui.keyUp("left")

        if y > %%:
            pyautogui.keyDown(%%)
        elif y < -%%:
            pyautogui.keyDown(%%)
        else:
            pyautogui.keyUp(%%)
            pyautogui.keyUp(%%)
```

✓ Ejercicio3

❖ Completa la función encargada del movimiento del micro:bit, sustituyendo los porcentajes (%) por los valores correctos en el programa move.py.

B. INTERACCIÓN

- La siguiente función en ser implementada es la interacción, basada en poder activar las animaciones de los objetos. En este caso, las animaciones son activadas mediante un evento de click sobre ellos, esta animación debe haber sido programada previamente y formó parte de una sección en el taller anterior de A-Frame.
- Para hacer esto con el micro:bit se tiene que "simular" un click al interactuar con él.
- La acción elegida en este ejemplo es presionar los dos botones a la vez (A, B).

✓Ejercicio 4

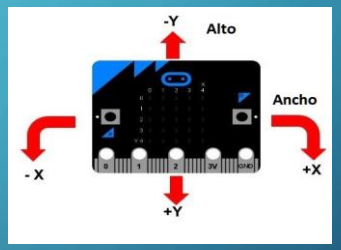
❖En el programa interaction.py y con la ayuda de las siguientes funciones de pyautogui, completa la función de interacción sustituyendo los porcentajes, recuerda que se debe hacer click cuando se presionan ambos botones del micro:bit.

- Hacer **click** -- pyautogui.click()
- Detectar la pulsación del botón **A** -- microbit.button_a.is_pressed()

```
1  import pyautogui
2  import microbit
3
4  #Function
5  def Interaction():
6      if %% and %%:
7          print("Button AB pressed, click mode")
8          %%
9
10 #MAIN
11
12 while True:
13
14     Interaction()
15
16     microbit.sleep(500)
```

B. CONTROL DE LA CÁMARA

- Para poder relacionar el movimiento de la cámara con el acelerómetro del Micro:bit, primero es necesario entender como se maneja la cámara en A-Frame con un ratón común. A la hora de mover la cámara se debe pulsar el botón izquierdo del ratón dentro del escenario de A-Frame y a continuación arrastrarlo en la dirección deseada.
- El objetivo es mapear el movimiento del ratón por la pantalla en base a los valores que ofrece el acelerómetro del micro:bit.
- Los pasos a seguir para implementar esta función deben ser los siguientes:



1. Extraer la resolución de la pantalla, puede ser extraída con la siguiente función (width/heigth). ***pyautogui.size()***
2. Guardar los valores del acelerómetro en el eje X e Y. ***microbit.accelerometer.get_x()***
3. Dejar pulsado el botón izquierdo del ratón para comenzar a manejar la cámara. ***pyautogui.mouseUp()***
4. Teniendo el movimiento de la cámara activo, el ratón debe desplazarse en función de los valores que proporciona el acelerómetro, siendo necesario multiplicar el valor actual del acelerómetro por el ancho y alto de la pantalla en el caso de mover el micro:bit a izquierda/derecha o arriba/abajo respectivamente.
5. Por último, se debe levantar la pulsación del botón izquierdo del ratón para salir del control de la cámara. ***pyautogui.mouseUp()***

✓ Ejercicio 5

- ❖ Implementa la función de control de la cámara en el programa camera.py siguiendo los puntos expuestos anteriormente.
- ❖ Añade las funciones anteriores al programa con el requisito de entrar en el modo de control de la cámara al pulsar B y salir de él al pulsar **B** de nuevo, por último, haz lo mismo con el botón **A** para el movimiento.