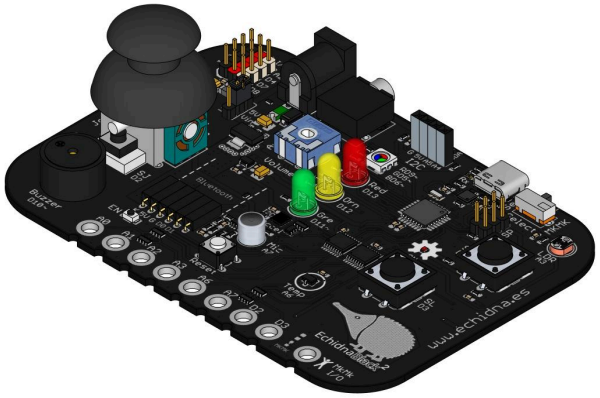
Manual EchidnaBlack^2 y EchidnaML

Índice:

1. ECHIDNABLACK Y ECHIDNAML

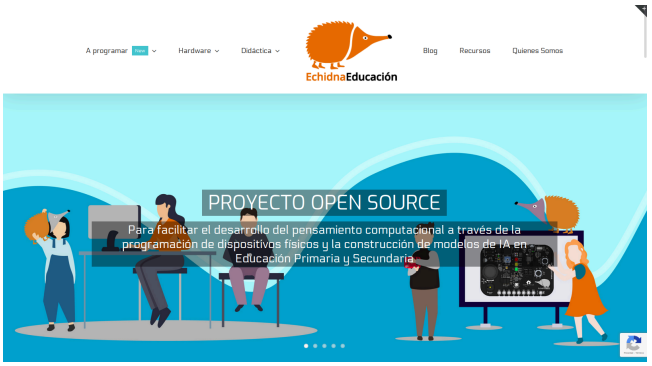
EchidnaBlack2 (hardware) y **EchidnaML** (software) constituyen un **sistema integrado**: están diseñados para aprender los fundamentos de la programación, la robótica y la Inteligencia Artificial (IA). Su **objetivo** principal es fomentar el desarrollo del **pensamiento computacional** en estudiantes de niveles básicos y avanzados (Primaria, Secundaria, Bachillerato y F.P.). Además, busca estimular la creatividad a través de la realización de proyectos y actividades prácticas que relacionan el mundo digital y el analógico.

El sistema se compone de una **placa** con diversos componentes integrados y un **entorno de programación** diseñado a medida para aprovechar todas sus posibilidades. Además, este conjunto se complementa con **materiales educativos** creados por docentes para facilitar el uso de la placa robótica y el software en el aula.



Esta **guía** ofrece una introducción al trabajo con la placa EchidnaBlack2 y su entorno de programación EchidnaML. No se requieren conocimientos de programación previos; sin embargo, se recomienda tener experiencia con el entorno visual de Scratch, ya que facilita la comprensión de la programación por bloques y el conocimiento del entorno.

Para ampliar y complementar la información ofrecida en esta guía, puede consultar recursos adicionales y material didáctico en la **web oficial del proyecto**: www.echidna.es.



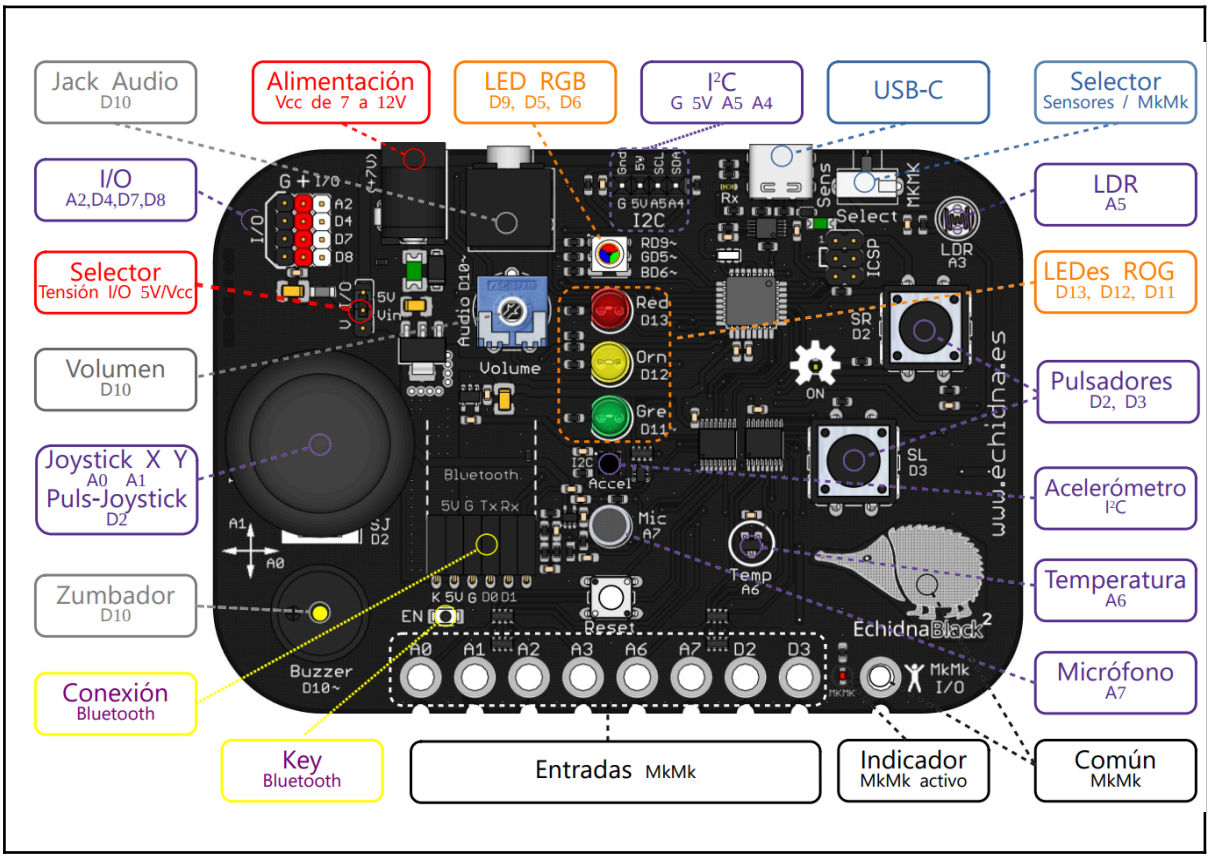
2. LA PLACA ECHIDNABLACK

La placa EchidnaBlack² es una placa autónoma basada en la arquitectura Arduino (es compatible con Arduino Nano y UNO).

La placa dispone de su propio entorno de programación por bloques, EchidnaML. Además, su arquitectura permite la integración con otros entornos de programación comunes en el ecosistema Arduino, tales como Arduino IDE, Snap4Arduino o mBlock. Esta versatilidad facilita el desarrollo de proyectos con distintos lenguajes y niveles de complejidad.

2.1 Componentes

La placa cuenta con los siguientes componentes:



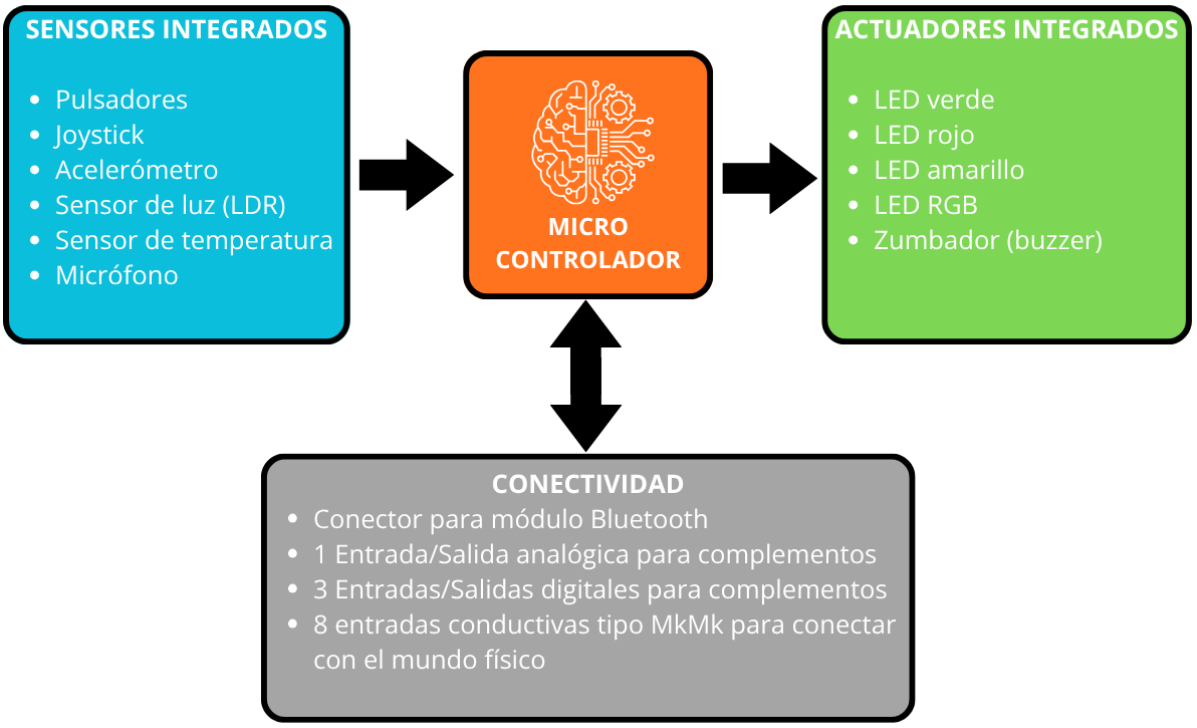
Las características de cada componente se verán en el apartado **"5- COMPONENTES Y BLOQUES DE PROGRAMACIÓN: DESCRIPCIÓN Y EJEMPLOS DE USO"**

Conexión a ordenador:

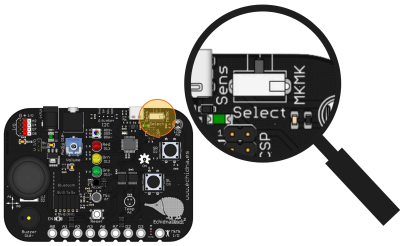
Para conectar la placa EchidnaBlack² al ordenador es necesario un cable USB-C.

En el apartado "**4.1 Puesta en marcha**" lo veremos con detalle.

Clasificación de los componentes de la placa según su función:

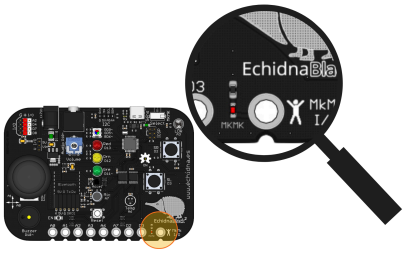


2.2 Modos de funcionamiento



La placa EchidnaBlack dispone de dos modos de funcionamiento, el **Modo Sensores** y el **Modo MkMk**. Mediante un conmutador se puede seleccionar en que modo queremos trabajar.

Modo Sensores: Recoge la lectura de todos los sensores que tiene la placa, pulsadores SR y SL, Joystick, sensor de luz (LDR), sensor de Temperatura Micrófono, Acelerómetro XYZ. y de todas las conexiones I/O, incluidas las I2C.



Modo MkMk: Nos da acceso a las 8 entradas conductivas tipo Makey Makey (MkMk) y al resto de las entradas no usadas en este modo.

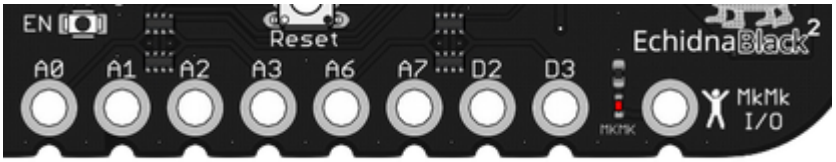
⚠️ **¡ATENCIÓN!** Un testigo luminoso rojo nos indica cuando estamos en modo MkMk.

2.3 Entradas MkMk

Las entradas tipo Makey Makey (MkMk) se activan cuando desplazamos el **conmutador** modos de funcionamiento a la derecha, y el LED rojo nos marca el testigo.

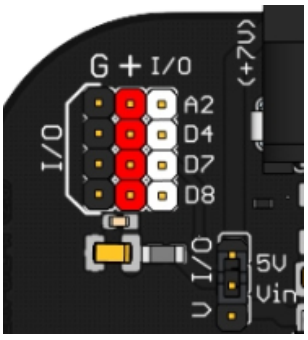
Las entradas MkMk son **capaces** de detectar conductividad eléctrica a través de una amplia variedad de materiales. La detección de la señal se produce cuando el circuito se cierra al tocar, de forma simultánea, el elemento conductor conectado a la entrada y el conector GND (o MkMk) de la placa.

La placa dispone de **8 entradas MkMk**: A0, A1, A2, A3, A6, A7, D2, D3.



Este modo **permite** crear proyectos que unen el mundo físico más cercano con el virtual que se desarrolla en el ordenador de forma muy sencilla y manipulativa, por ejemplo mediante pulsadores hechos con plastilina conductiva, frutas, papel de aluminio o agua del grifo, añadiendo un componente lúdico, manipulativo y de exploración a los proyectos

2.4 Pines de entrada/salida



Los pines de entrada/salida (I/O) son conexiones que nos permiten conectar dispositivos adicionales a la placa, actuando como entradas (sensores) o salidas (actuadores).

Cuando funcionan como entrada reciben información de sensores externos. Cuando funcionan como salida, envían señales desde la placa para controlar actuadores externos.

La placa EchidnaBlack² cuenta con tres pines de entrada/salida digitales (D4, D7, D8) y uno analógico (A2), a los que podemos conectar una gran variedad de componentes, como servomotores, sensores de distancia infrarrojos, sensores de humedad de suelo, etc.

Cada pin de entrada/salida cuenta con una conexión a 0 V (G), 5 V (+), y el pin de señal (I/O).

Cerca de estos pines de entrada/salida se encuentra un **selector de alimentación** que permite elegir de dónde reciben energía estos pines moviendo un jumper entre dos posiciones:

- En la **posición 5V**, la corriente procede del regulador interno y da una tensión estable adecuada para componentes que no requieran mucha potencia, ya que no debe utilizarse si estos componentes superan los 300 mA de consumo, ya que se sobrepasaría el regulador.

- En la **posición Vin**, la alimentación llega desde una fuente externa conectada al jack, lo que resulta más seguro cuando se utilizan actuadores que requieren mayor potencia.

2.5 Alimentación

Alimentación por USB-C

La alimentación a través de USB-C es la forma más recomendable para la **mayoría** de los **usos**.

Esta es la forma más sencilla, y común de usar la placa, mediante un cable USB-C conectado a un ordenador. La placa recibe una tensión de 5 V y suficiente energía para las funciones básicas.

Alimentación por Jack

La usamos cuando queremos conectar elementos con mucha potencia, como varios servomotores, o para proyectos autónomos.

3. ECHIDNAML

EchidnaML es una programa de **escritorio** que integra una versión de **Scratch** que proporciona nuevos bloques para programar las placas Echidna e integra el módulo de inteligencia artificial **LearningML**.

Así que en un mismo entorno podemos trabajar con Scratch, robótica e inteligencia artificial.

Para empezar a trabajar con EchidnaML, es necesario descargar el programa desde la página web de Echidna Educación: https://echidna.es/a-programar/echidnaml/

Pantalla de inicio:

En la pantalla de inicio de EchidnaML tenemos dos iconos uno para elegir EchidnaScratch para empezar a utilizar Scratch y controlar la placa o LearningML para crear modelos de aprendizaje automático.

En la **pantalla de inicio** podemos encontrar:

1. Versión del programa EchidnaML.
2. Si el programa está conectado a nuestra placa y la versión de nuestra placa.
3. Puerto (USB) de conexión: podemos ver el puerto USB al que se conecta nuestra placa, en el caso de que conectemos la placa posteriormente al inicio del programa podemos pulsar en el icono para que se reconecte.
4. Configuración de idioma: nos permite seleccionar el idioma
5. Acceso directo al programa EchidnaScratch
6. Acceso directo al programa LearningML
7. Botón para salir del programa

Trabajo sin la placa conectada:

EchidnaML permite que trabajemos sin conectar la placa. Sin la placa conectada podemos:

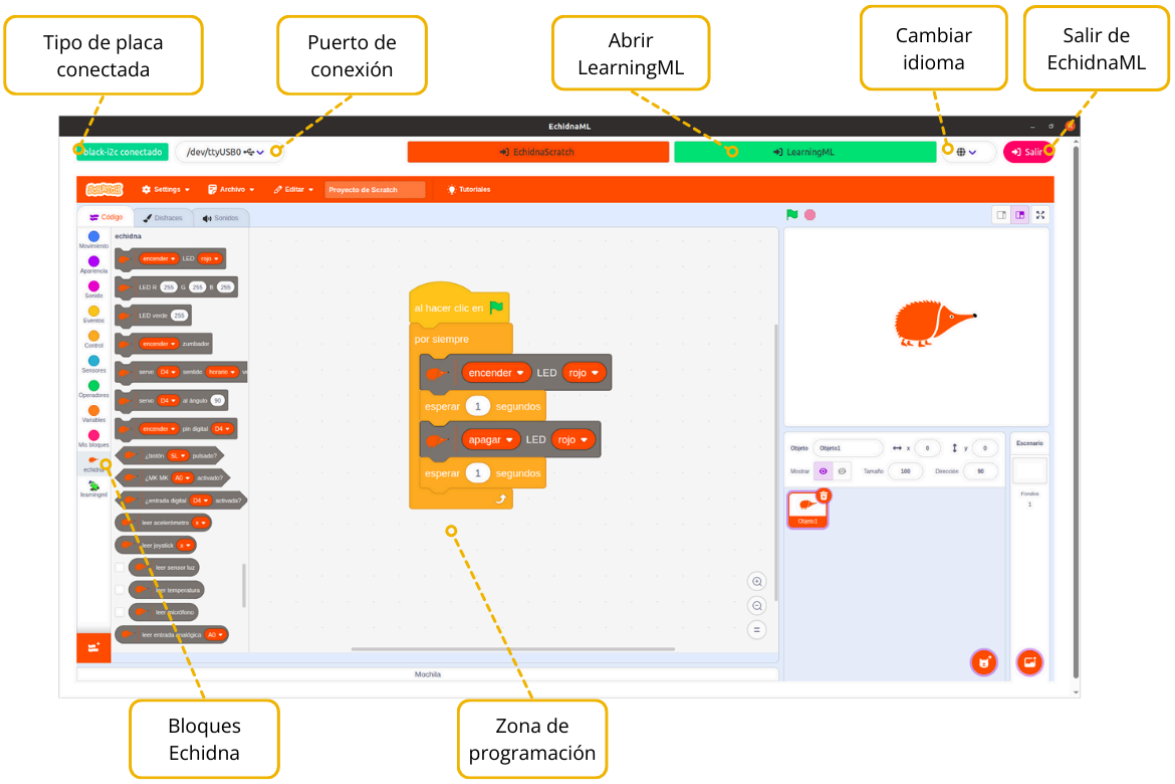
1. Usar LearningML para construir modelos
2. Usar Scratch
3. Programar un programa de robótica y guardarlo en el equipo para cuando conectemos la placa

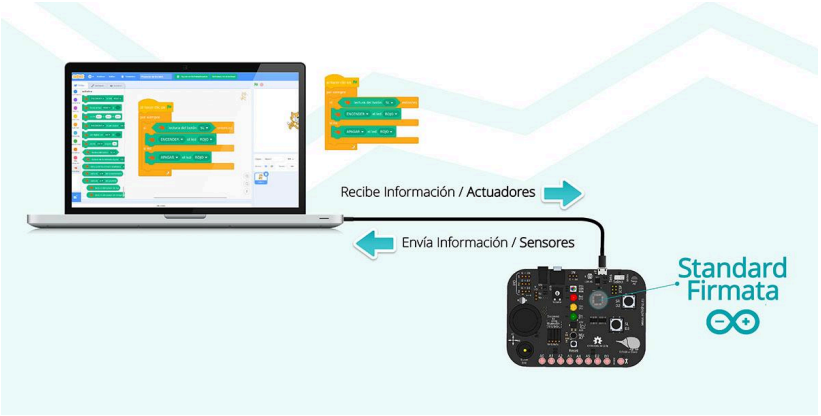
Reconectar la placa con el programa abierto:

Si reconecto la placa una vez abierto EchidnaML se perderá todo el trabajo realizado por lo que debemos guardarlo antes para poder recuperarlo.

4. ECHIDNASCRATCH

EchidnaScratch es una versión de Scratch a la que se le han añadido bloques para controlar la placa EchidnaBlack y para incorporar Machine Learning a través de los bloques de Learning ML.





Comunicación placa-programa: EchidnaScratch se comunica con el microcontrolador a través del puerto serie. La placa envía información sobre el estado de los sensores y el programa que realizamos en EchidnaScratch lo procesa y devuelve información sobre el estado de los actuadores de la placa.

Bloques de programación de la placa

EchidnaScratch ha **añadido** los siguientes **bloques** de programación a Scratch que permiten **controlar** el funcionamiento de la placa. Leyendo la información de los **sensores**, y mandando señales para controlar los **actuadores**.

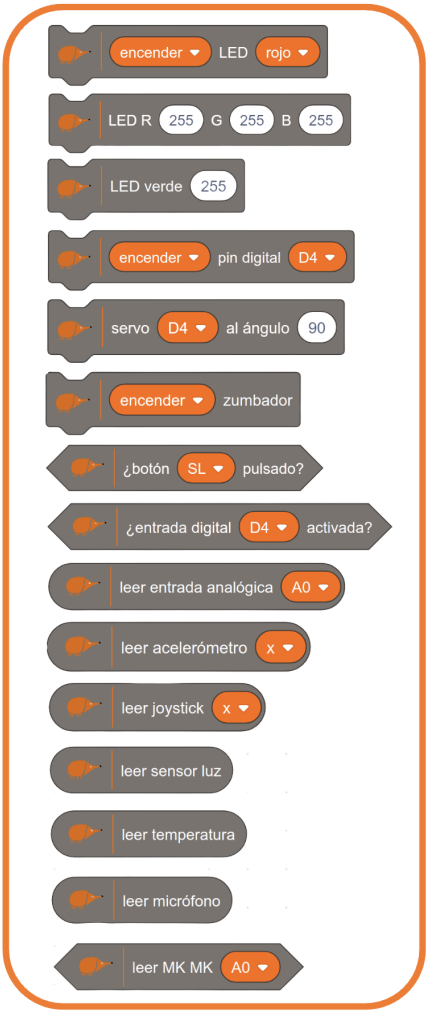
Si te fijas, por la forma, hay 3 tipos de bloques:

Rectangulares: son bloques dedicados a programar los **actuadores**, LED, Zumbador, servo…

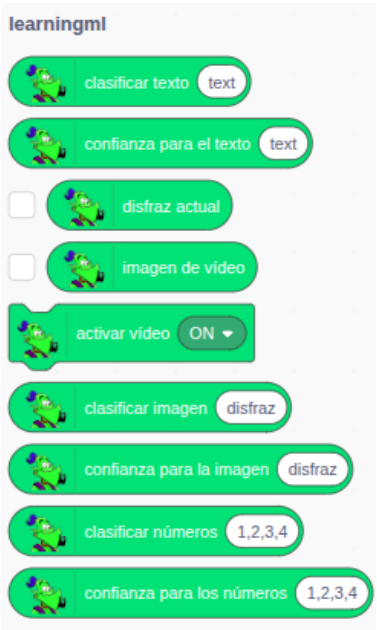
Trapezoidales: son bloques destinados a leer **sensores**, entradas digitales, que devuelve un *True* o *False*.

Redondeados: son bloques destinados a leer **sensores**, entradas analógicas, que devuelven el valor del sensor.

No te preocupes ahora por lo que hace cada uno de los bloques, lo veremos en el siguiente apartado a través de ejemplos.



Bloques de programación de LearningML



EchidnaML integra **LearningML**, una plataforma educativa diseñada para la enseñanza de los fundamentos del Machine Learning (aprendizaje automático). Esto nos permite incorporar Inteligencia Artificial a nuestros proyectos de robótica.

EchidnaScratch incorpora bloques específicos de **Machine Learning (ML)** que permiten construir aplicaciones de **Inteligencia Artificial (IA)** capaces de reconocer imágenes, números y textos escritos en lenguaje natural.

Estos bloques, que se encuentran en las sección **LearningML**, pueden combinarse con los bloques de control

de las placas Echidna y con las funcionalidades estándar de Scratch.

De esta forma, los estudiantes pueden **unir el mundo de la IA con la robótica educativa** para crear proyectos avanzados.

Para usar los bloques de Machine Learning, primero debes crear un modelo de reconocimiento de imágenes o textos. Pero eso lo contaremos en la sección 6 de este manual.

4.1 Puesta en marcha

Pasos para comenzar a usar EchidnaML y EchidnaBlack:

1- Conecta la placa Echidna al ordenador mediante el cable USB C.

Las placas Echidna tienen cargado de fábrica el programa StandarFirmata, necesario para la comunicación con el ordenador a través del puerto serie (USB). Si lo has borrado de tu placa, visita la sección 6 de este manual.

2- Abre el programa EchidnaML

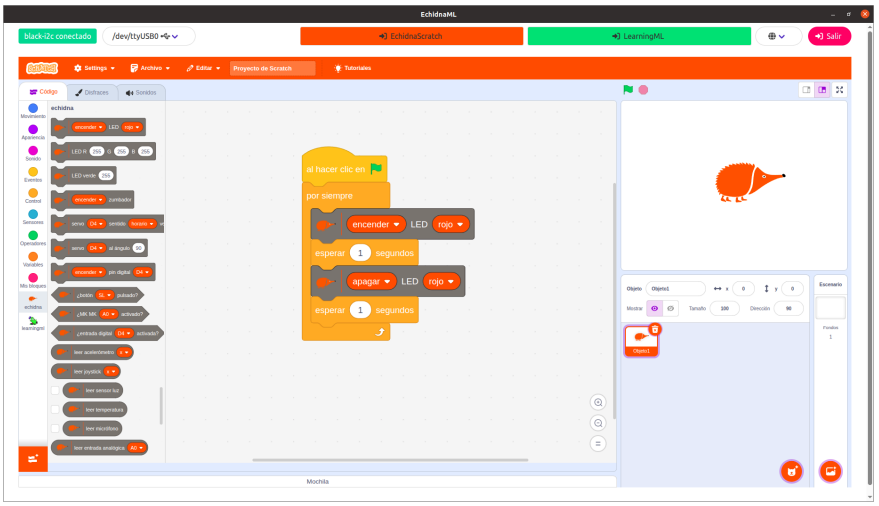
Clica en el icono de EchidnaML para abrir el programa. Recuerda que previamente debes haber instalado el programa.

3- Abrir EchidnaScratch

Al iniciar el EchidnaML, este se conecta automáticamente con la placa. Cuando lo haga aparecerá un aviso indicando la conexión.

Para acceder a EchidnaScratch, haz clic en el icono o en el botón Abrir EchidnaScratch. Esto llevará al programa, donde encontraremos los bloques habituales de Scratch, y los especiales para programar los dispositivos de la placa Echidna.

4- ¡Empieza a programar!



Hola Mundo

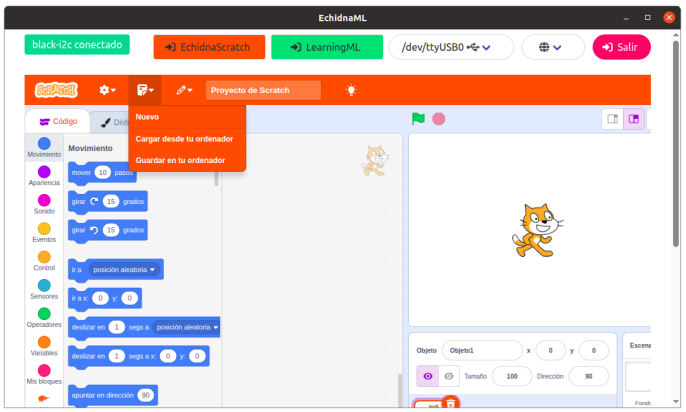
Combinando estos nuevos bloques con los clásicos de Scratch podremos programar de forma sencilla dispositivos físicos.

¿Hacemos un *Hola, mundo*?

Prueba a programar un LED intermitente utilizando los bloques de la imagen.

4.2 Guardar e Importar proyectos en EchidnaScratch

EchidnaScratch es un programa de **Escritorio,** por lo que si queremos **guardar** nuestros proyectos debemos descargarlos y almacenarlos en nuestro ordenador. Para ello debemos pulsar en *Archivo/ Guardar en tu ordenador* y almacenarlo en la carpeta que elijamos.



Para **recuperar** el proyecto debemos pulsar en *Archivo/ Cargar desde tu ordenador* y buscar el archivo en la carpeta donde lo almacenamos.

Los proyectos se almacenan en formato sb3, que es el mismo tipo de ficheros que Scratch.

EchidnaScratch es **compatible** con los proyectos realizados en **Scratch,** por lo que puedes descargar cualquier proyecto realizado en

Scratch y cargarlo en EchidnaScratch dotándolos de interactividad a través de los sensores de la placa.

5. COMPONENTES Y BLOQUES DE PROGRAMACIÓN: DESCRIPCIÓN Y EJEMPLOS

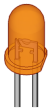
En este apartado vamos a revisar los componentes que integra la placa, viendo las principales **características** que tienen, los **bloques** de **programación** que podemos usar para controlarlos, y un ejemplo de su uso en el entorno de programación.

5.1 Componentes de la placa

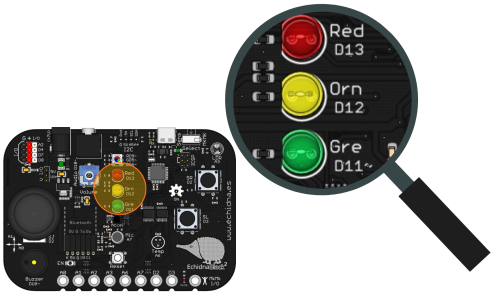
A continuación se describen los componentes que lleva integrados la placa.

5.1.1 Ledes: Semáforo e intensidad luminosa

⚙️ **COMPONENTE:**



Diodo **LED**: es el acrónimo de Light Emitting Diode (Diodo emisor de luz), está basado en el fenómeno de electroluminiscencia. Se usan como testigos (indicadores) y como fuente de iluminación.

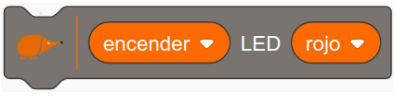


En la placa tenemos 3 diodos LED Verde, Naranja y Rojo.

Los LEDs Naranja y Rojo pueden ser controlados únicamente de forma digital (estados de encendido/apagado). Por otro lado, el LED Verde puede ser controlado digitalmente y, además, permite el control de su intensidad luminosa (modulación analógica o PWM).

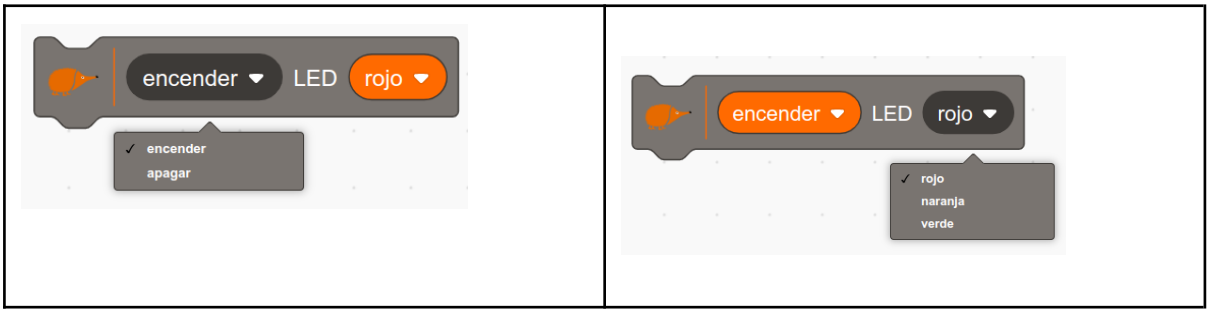
🖥️ **BLOQUE DE PROGRAMACIÓN:**

Para controlar **digitalmente** los LEDs podemos usar el siguiente bloque:

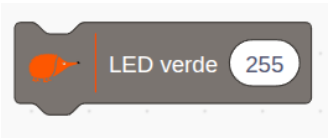


En el que podemos:

Controlar el **estado**: encender o apagar:	Controlar **qué LED** queremos encender/apagar:



Para controlar la intensidad luminosa del **LED Verde** podemos usar el siguiente bloque:



En el que podemos regular la intensidad luminosa entre 0 (apagado) y 255 (máxima intensidad luminosa)

🛠️ **EJEMPLO**: Semáforo

Este ejemplo muestra cómo programar un semáforo utilizando los LEDs de la placa. La programación se basa en una secuencia cíclica donde cada LED permanece encendido durante un tiempo específico y luego pasa al siguiente estado de forma automática.

Estados:

1. El LED verde se enciende durante 5 segundos. Al finalizar este tiempo, se apaga.
2. El LED naranja se enciende durante 2 segundos, y luego se apaga.
3. El LED rojo se enciende durante 5 segundos. Transcurrido este tiempo, se apaga.

Y el ciclo vuelve a comenzar con la Luz Verde, remitiéndose de forma indefinida.

🛠️ **EJEMPLO**: Control intensidad luminosa LED Verde

Este ejemplo muestra cómo controlar la intensidad luminosa del LED verde para crear un efecto de "fundido" (fade) gradual, simulando un ciclo suave de encendido y apagado.

Para lograrlo, el programa utiliza dos bucles secuenciales que modifican progresivamente el valor de intensidad del LED (de 0 a 255):

Bucle de Encendido Progresivo: La intensidad del LED aumenta de 0 (apagado) hasta el valor máximo (255) en pasos pequeños, logrando un encendido suave.

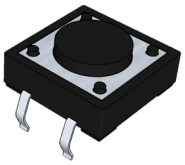
Bucle de Apagado Progresivo: La intensidad del LED disminuye de 255 (máxima luminosidad) a 0, creando un efecto de fundido hasta apagarse por completo.

Ambos bucles se ejecutan consecutivamente para generar el efecto de fundido suave y cíclico. Volviendo a repetirse al final del ciclo.

5.1.2 Pulsadores: Encender/Apagar Led

⚙️ **COMPONENTE:**

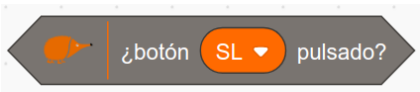
El pulsador es un componente electromecánico que permite abrir o cerrar un circuito con un solo estado estable.



Tenemos 2 pulsadores, SL (Switch Left) y SR (Switch Right), situados en la parte derecha de la placa.

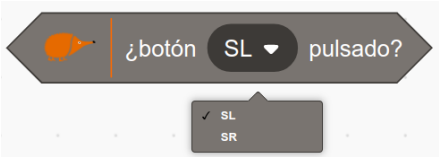
🖥️ **BLOQUE DE PROGRAMACIÓN:**

Para leer el estado del pulsador podemos usar el siguiente bloque:

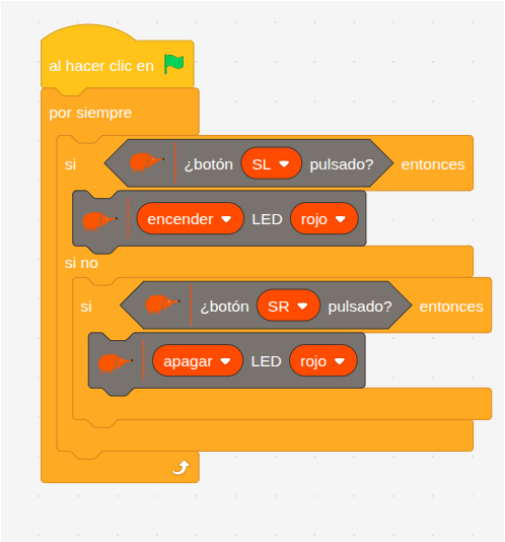


Cuando leemos el estado del pulsador nos devuelve **true** (1) si está pulsado y **false** (0) si no está pulsado.

El bloque nos permite seleccionar el pulsador derecho o el izquierdo.



🛠️ EJEMPLO:



Este ejemplo utiliza dos pulsadores para controlar el encendido y apagado del LED rojo. El pulsador izquierdo (SL) controla el encendido del LED rojo y el pulsador derecho (SR) el apagado.

Funcionamiento:

El programa comprueba primero: SI el pulsador izquierdo (SL) está presionado.

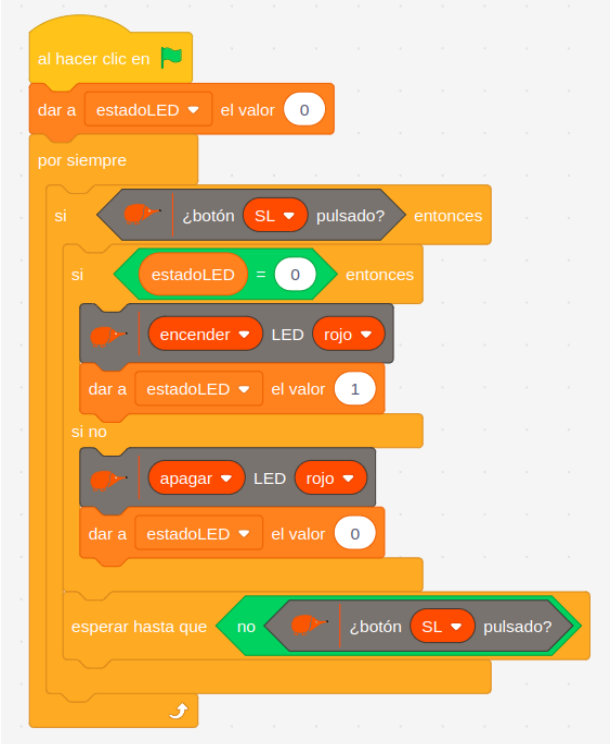
ENTONCES enciende el LED Rojo inmediatamente

SI NO (es decir, si SL no está presionado), el programa comprueba la segunda condición.

SI el pulsador derecho (SR) está presionado.

ENTONCES apaga el LED Rojo

🛠️ EJEMPLO: Pulsador con memoria



Este ejemplo ilustra cómo programar un pulsador para que actúe como un interruptor. Es decir, la primera pulsación enciende un LED y la siguiente pulsación lo apaga.

Para que el sistema pueda "recordar" si el LED está actualmente encendido o apagado, utilizamos una variable de estado denominada *estadoLED*.

De esta forma, la variable *estadoLED* actúa como la memoria que permite al pulsador alternar entre los dos estados con cada activación.

Funcionamiento:

1. Cuando se pulsa el botón, el programa consulta el valor actual de la variable *estadoLED*.
2. Si *estadoLED* indica APAGADO (valor 0), el programa:
    - Enciende el LED.
    - Cambia el valor de estadoLED a ENCENDIDO (valor 1).
3. Si no (si *estadoLED* indica ENCENDIDO valor 1), el programa:
    - Apaga el LED.
    - Cambia el valor de estadoLED a APAGADO (valor 0).

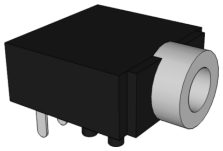
Para asegurar que el cambio de estado (encendido → apagado, o viceversa) ocurra solo una vez por cada activación, es esencial incluir un mecanismo que evite que el programa ejecute la condición SI pulsado múltiples veces mientras el botón se mantiene presionado.

Para ello, se añade un bloque "esperar hasta que" que mantiene el programa en pausa hasta que el pulsador sea liberado. Esta técnica previene el efecto de rebote y garantiza que la variable de estado se altere una sola vez por cada pulsación física, logrando un control de memoria estable.

5.1.3 Zumbador: pulsador-sonido

⚙️ COMPONENTE:

Disponemos de dos salidas para reproducir audio, el zumbador y el jack al que podemos conectar auriculares o altavoces autoamplificados. Al conectar una clavija de audio en el jack se desconecta el zumbador.
Además, contamos con un potenciómetro que permite ajustar el volumen del sonido.



🖥️ BLOQUE DE PROGRAMACIÓN:

Para controlar el zumbador podemos usar el siguiente bloque:

🛠️ EJEMPLO:

Este ejemplo muestra cómo controlar el zumbador (actuador de sonido) utilizando un pulsador como entrada.

- Si el pulsador es presionado (o activado), el zumbador suena.

- Si el pulsador es liberado (o soltado), el zumbador deja de sonar.

De esta forma, el zumbador solo se activa mientras el pulsador se mantiene presionado.

5.1.4 Sensor de Luz (LDR): Interruptor crepuscular

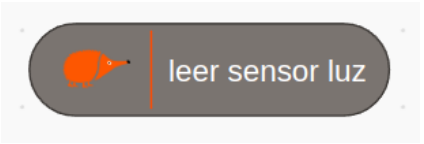
⚙️ COMPONENTE:

LDR es el acrónimo de "Light Dependent Resistor", es una resistencia cuyo valor depende de la cantidad de luz que incide sobre ella.

En la placa Echidnablack² podemos encontrar la LDR en la esquina superior derecha.

🖥️ BLOQUE DE PROGRAMACIÓN:

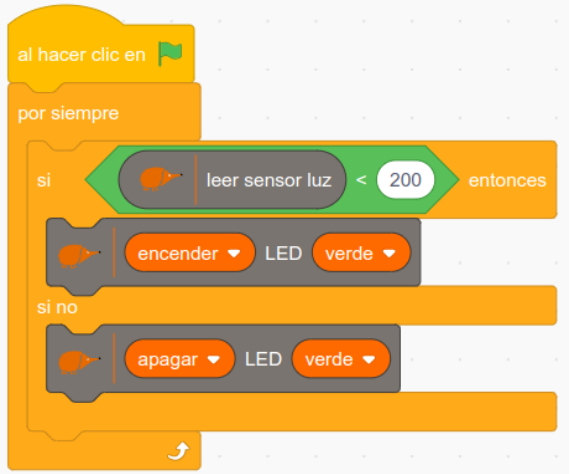
Para leer el valor del sensor podemos usar el siguiente bloque:



Puedes activar la casilla de verificación para ver el valor registrado.

El sensor de luz proporciona valores bajos con poca luz y valores altos con mucha luz. Con una variabilidad entre 0 (no hay luz) y 1023 (mucha luz).

🛠️ EJEMPLO:



Este ejemplo implementa un **interruptor crepuscular**, permitiendo que un LED se encienda automáticamente en función de la intensidad de la luz ambiental detectada por la LDR.

Programamos la placa para que:

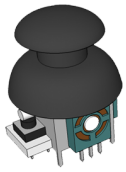
Si el sensor de luz registre valores menores de 200 → se encienda el LED verde.

Si no (si registra valores mayores) → se apague.

El valor 200 actúa como el umbral que define cuándo debe encenderse o apagarse la luz.

5.1.5 Joystick: Pintamos

⚙️ COMPONENTE:



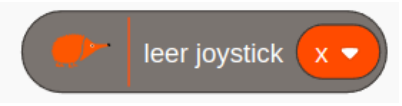
Es un dispositivo de entrada con una palanca que se mueve en varias direcciones. Al inclinarla, el joystick envía señales que indican hacia dónde y cuánto se ha desplazado respecto a dos ejes: horizontal (X) y vertical (Y).

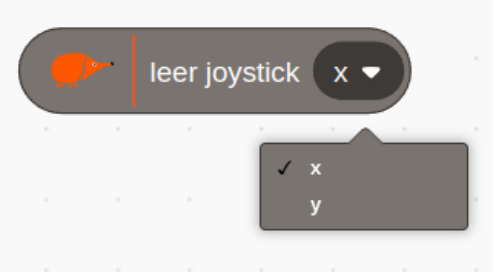
Consta internamente de dos potenciómetros, uno para el eje X (movimiento horizontal) y otro para el eje Y (movimiento vertical), que convierten la posición del stick en valores de resistencia.

Pulsador adicional: este modelo incorpora además un pulsador (botón) que se activa al presionar el stick hacia abajo. En Echidnablack², el pulsador del Joystick está conectado directamente con el pulsador "SR".

🖥️ BLOQUE DE PROGRAMACIÓN:

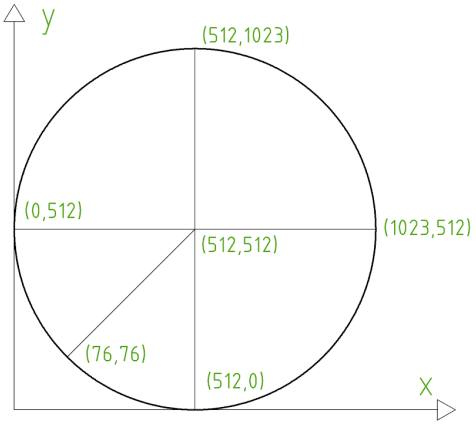
Para leer el valor del joystick podemos usar el siguiente bloque:





En el bloque seleccionamos eje x, eje y.

A la derecha podemos ver una gráfica de los valores proporcionados por el joystick.



Valores proporcionados por el joystick:

- El joystick en reposo proporciona valores en torno a 512 en el eje x e y.
- En el eje X proporciona valor de 0 a la izquierda y valor 1023 a la derecha.
- En el eje Y valor 0 abajo y 1023 arriba.

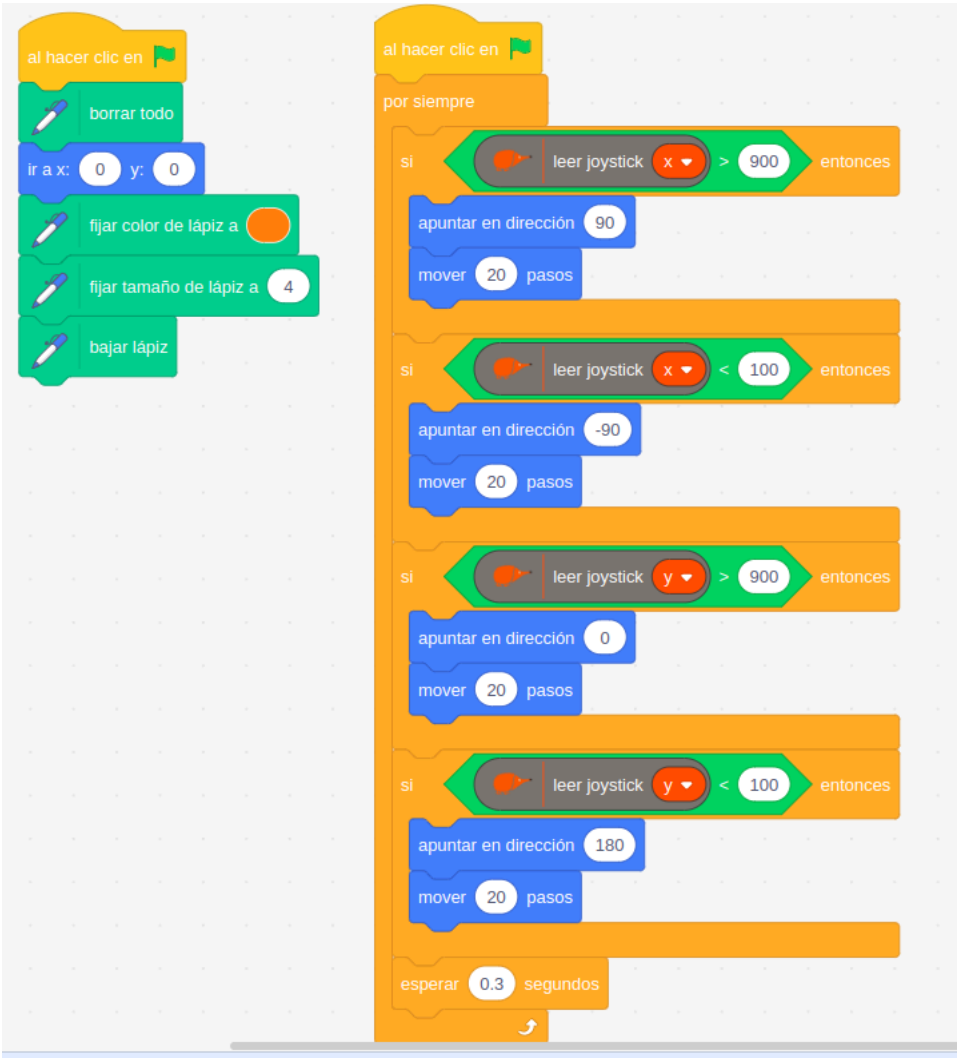
🛠️ EJEMPLO:

Este ejemplo utiliza el Joystick (control de dos ejes) para controlar el movimiento de un puntero o lápiz virtual en la pantalla del entorno de programación (EchidnaML).

1. **Cargar la herramienta lápiz:** Lo primero que tenemos que hacer es cargar la herramienta lápiz.
2. **Configuración inicial:** Configuramos la herramienta lápiz para que al inicio:
    - Borre el fondo

- Se vaya al punto (0,0)
- fijamos el color y el grosor con el que se va a pintar
- Bajamos el lápiz

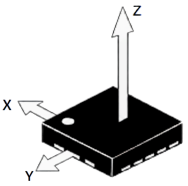
3. **Lógica de movimiento, control del joystick:** Programamos la placa para que:
   A. Cuando el joystick se desplace a la derecha y registre valores menores de 900 en el eje x, el puntero se desplace hacia la derecha.
   B. Cuando se desplace a la izquierda y registre valores menores de 100 en el eje x, el puntero se desplace hacia la izquierda.
   C. Cuando el joystick se desplace hacia arriba y registre valores mayores de 900 en el eje y, el puntero se desplaza hacia arriba.
   D. Cuando se desplace a abajo y registre valores menores de 100 en el eje y, el puntero se desplace hacia abajo.



5.1.6 Acelerómetro: Movemos el erizo

⚙️ COMPONENTE:

Es un sensor microelectromecánico (MEMS) de aceleración que mide los movimientos en los tres ejes: X, Y y Z.

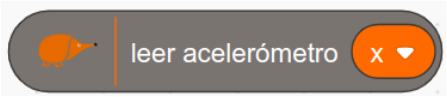


Mide la inclinación en los ejes **X e Y**. Esto es posible porque la aceleración de la gravedad (1g) genera un cambio de capacitancia proporcional al ángulo de inclinación del sensor

Permite detectar cambios bruscos o dinámicos de movimiento en el eje Z. Cualquier movimiento vertical repentino provoca una variación rápida en la aceleración medida a lo largo de este eje.

🖥️ BLOQUE DE PROGRAMACIÓN:

Para leer el valor del joystick podemos usar el siguiente bloque:



En el bloque seleccionamos eje x, eje y o eje z.

Valores:

- El acelerómetro en reposo proporciona valores en torno a 0 en los ejes x e y, y 1 en el eje z.
- En el eje x proporciona valor de 0 a -1 al elevar la parte derecha y de 0 a 1 al elevar la parte izquierda.
- En el eje y proporciona valor de 0 a -1 al elevar la parte trasera y de 0 a 1 al elevar la parte delantera.
- En el eje z se pueden registrar valores de menores de -2 al subirla placa rápidamente y de más de 2 al bajarla rápidamente.

🛠️ **EJEMPLO:** Gatito atleta.

Programamos la placa en dos hilos de ejecución:

Hilo de control de movimiento:

SI la placa se inclina a la izquierda y registra valores menores de -0,5 en el *eje x*, el gato se desplace hacia la izquierda.

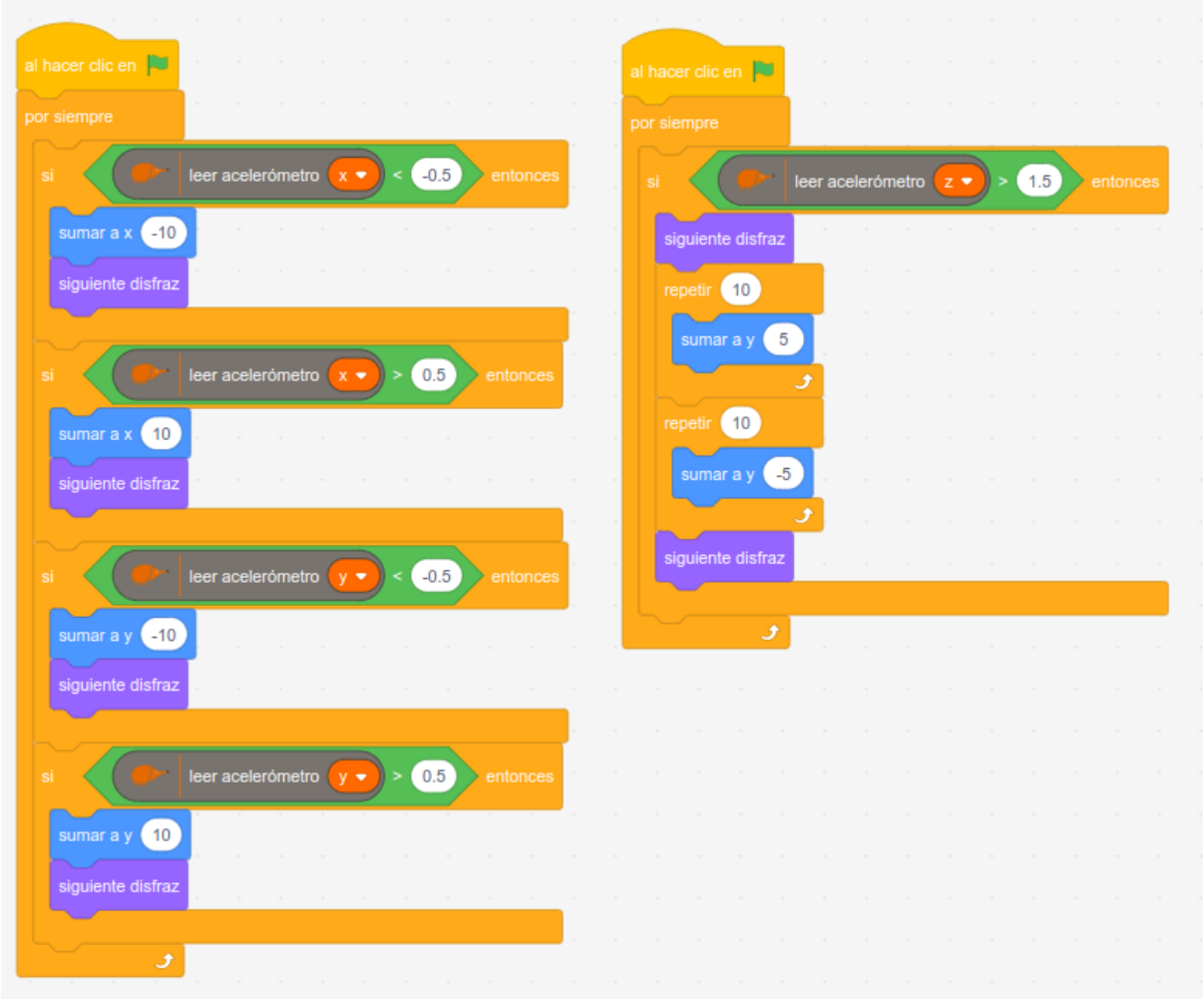
SI la placa se inclina a la derecha y registra valores mayores de 0,5 en el *eje x*, el gato se desplace hacia la derecha.

SI la placa se inclina hacia atrás y registra valores menores de -0,5 en el *eje y*, el gato se desplace hacia abajo.

SI se inclina hacia delante y registra valores mayores de 0,5 en el *eje y*, el gato se desplace hacia arriba.

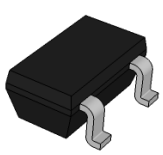
Hilo de control de salto:

Cuando se eleve bruscamente la placa y el *eje z* registre valores mayores de 1.5, el gato efectuará un salto.



5.1.7 Sensor de temperatura: El erizo dice la temperatura

⚙️ **COMPONENTE:**



El MCP9700T es un sensor que entrega un voltaje analógico proporcional a la temperatura en grados Celsius (°C).

- Sensibilidad: Cada 10 mV equivalen a 1°C.

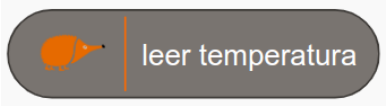
- Offset: Posee un voltaje de 500 mV (0.5V) a 0°C.

Fórmula de Conversión: La temperatura en grados Celsius se calcula como:

$$T(°C)=(V-0.5)\times 100.$$

🖥️ BLOQUE DE PROGRAMACIÓN:

Para leer la temperatura podemos usar el siguiente bloque.

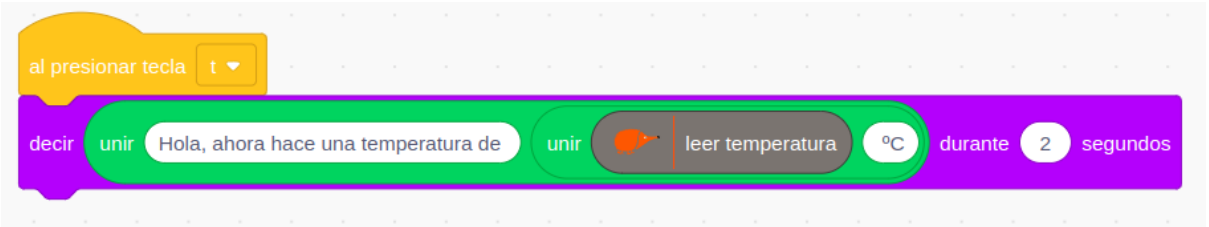


El bloque utiliza la fórmula de conversión anterior para convertir la tensión de lectura en voltios a ºC.

Puedes activar la casilla de verificación para ver el valor registrado.

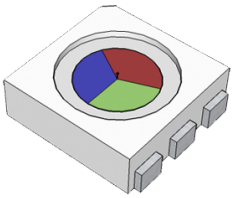
🛠️ EJEMPLO:

Cuando pulsamos la letra t, el Gato nos dice que temperatura hace.



5.1.8 LED RGB: Mezclamos colores

⚙️ COMPONENTE:



El LED RGB es un único componente que integra tres diodos emisores de luz (LEDes) independientes (Rojo, Verde y Azul) dentro de la misma cápsula.

El acrónimo significa Light Emitting Diode (Diodo Emisor de Luz) y Red Green Blue (Rojo, Verde, Azul).

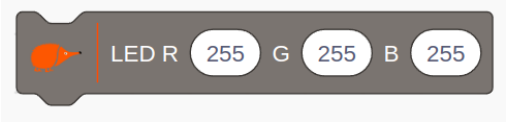
Control de Luminosidad LED (PWM): La luminosidad de cada uno de los tres LEDs se puede ajustar individualmente utilizando el control PWM (Modulación por Ancho de Pulso), a menudo denominado "analógico" en este contexto.

Generación de Colores: Podemos controlar el brillo de cada color (R, G y B) variando su intensidad desde 0 (apagado) hasta 255 (máxima intensidad).

Capacidad Cromática: Dado que cada canal ofrece 256 niveles de intensidad, la combinación de los tres colores (Rojo, Verde y Azul) permite generar un total de 256 x 256 x 256 = 16 777 216 (más de 16 millones) de colores diferentes.

🖥️ BLOQUE DE PROGRAMACIÓN:

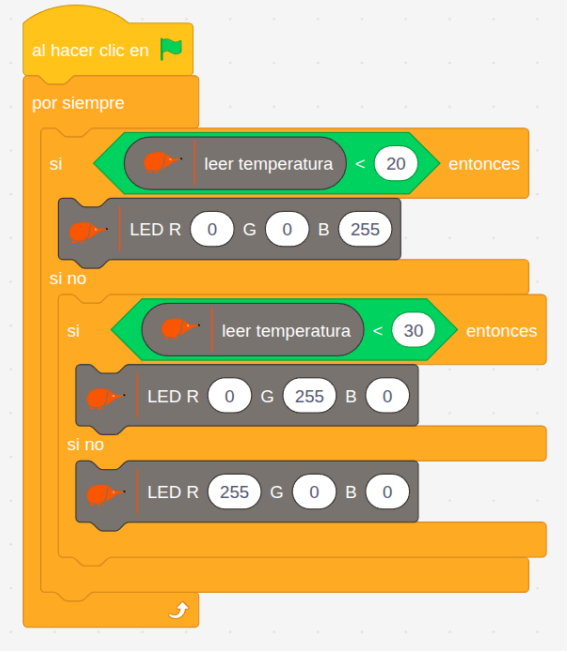
Para controlar la luminosidad y el color del LED RGB tenemos el siguiente bloque:



En el que podemos ajustar el valor de cada LED entre 0 y 255.

Así si queremos reproducir el naranja Echidna podemos poner:

🛠️ EJEMPLO: Indicador de Temperatura RGB



Este ejemplo utiliza el sensor de temperatura como dato de entrada para controlar el color del LED RGB (actuador). El objetivo es que el LED cambie de color automáticamente para indicar visualmente si la temperatura ambiente es baja (Azul), media (Verde) o alta (Rojo), funcionando como un termómetro visual.

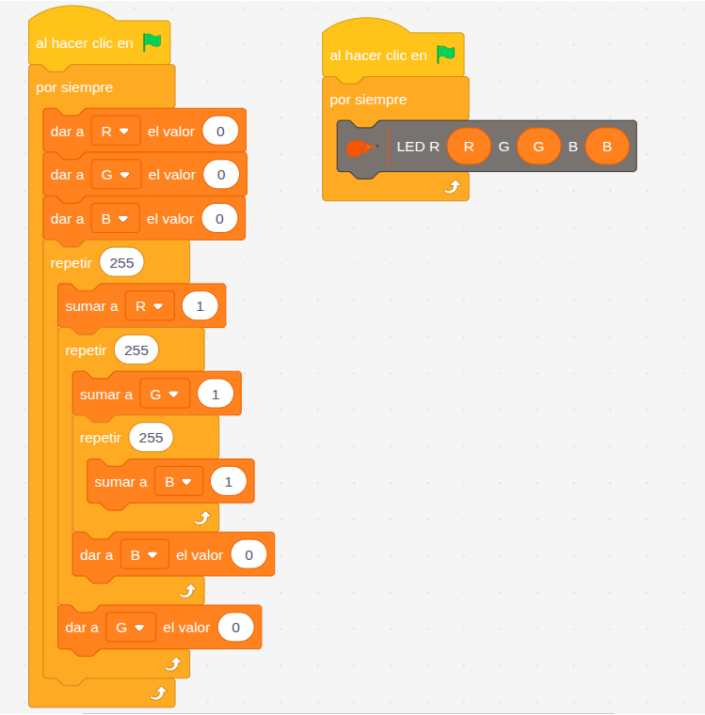
El programa evalúa continuamente la temperatura ambiente y asigna un color específico según el umbral que se cumpla:

Zona Fría (Alerta Azul): SI la temperatura es inferior a 20°C, el LED se ilumina en AZUL.

Zona Media (Temperatura Óptima/Verde): SI NO, SI la temperatura está entre 20°C y 30°C, el LED se ilumina en VERDE.

Zona Caliente (Alerta Roja): SI NO (es decir, si la temperatura supera los 30°C), el LED se ilumina en ROJO.

🛠️ **EJEMPLO:** Recorremos los 16.7 millones de colores



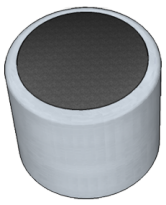
Este programa utiliza tres bucles anidados de repetición para recorrer sistemáticamente la totalidad de las combinaciones de color. Al variar la intensidad de cada canal (Rojo, Verde y Azul) en sus 256 niveles posibles, el sistema es capaz de generar los más de 16.7 millones de colores únicos que componen el espectro RGB.

256×256×256≈16.7 millones de colores

.

5.1.9 Micrófono: Vúmetro

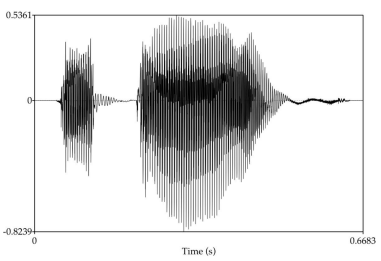
⚙️ **COMPONENTE:**



Es un transductor acústico-eléctrico. Utiliza el efecto piezoeléctrico para convertir las vibraciones de sonido en una señal eléctrica.

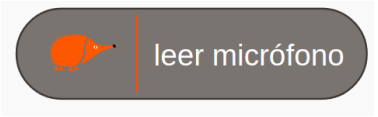
Principio de funcionamiento: Al recibir una onda sonora, el material piezoeléctrico genera una señal eléctrica que reproduce las mismas características (frecuencia y amplitud) del sonido captado.

Variabilidad de la señal: La señal eléctrica refleja directamente el sonido recibido, por lo que presenta una gran variabilidad. Se trata de una señal analógica compleja y que cambia constantemente, por lo que para poder trabajar adecuadamente con ella requiere un **procesamiento** posterior para su análisis (por ejemplo hallando la media aritmética), o, como alternativa, puede limitarse a detectar únicamente la intensidad del sonido.



🖥️ BLOQUE DE PROGRAMACIÓN:

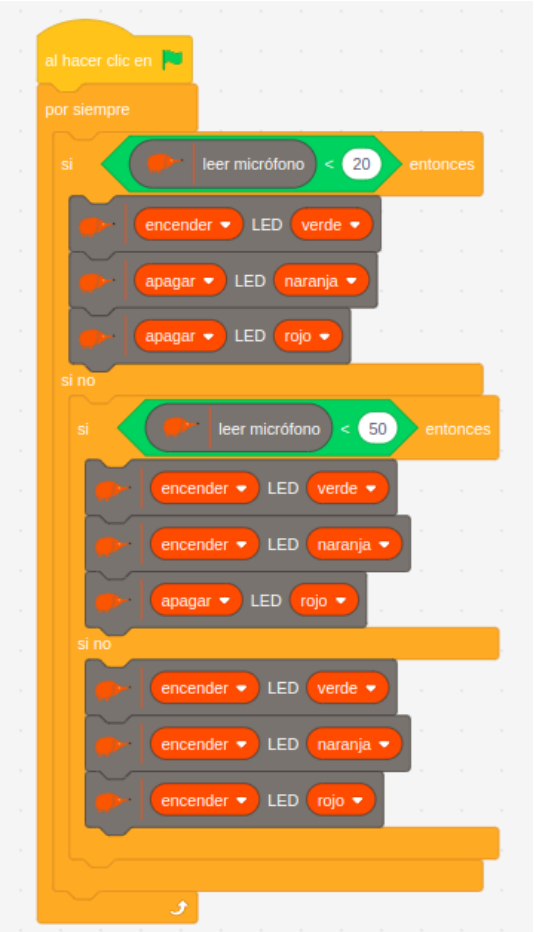
Para leer el valor del sensor podemos usar el siguiente bloque:



Puedes activar la casilla de verificación para ver el valor registrado.

Valores: el micrófono proporciona valores bajos en presencia de poco sonido o silencio, y valores más altos cuando capta sonidos intensos. Con una variabilidad entre 0 (no hay sonido) y 1023 (sonido de alta intensidad).

🛠️ EJEMPLO:



Este ejemplo programa la placa para actuar como un semáforo de ruido o vúmetro, que indica visualmente la intensidad del sonido ambiente. El sistema opera según tres umbrales de sonido:

Nivel Bajo: Si el sensor de sonido registra valores menores a 20 (ambiente silencioso), solo se enciende el LED verde.

Nivel Medio: Si los valores están **entre 20 y 50**, se encienden los LEDs **verde y naranja** (precaución).

Nivel Alto: Si el valor es **superior a 50**, se encienden los **tres** LEDs (alerta).

Es probable que, al ejecutar este código, se observe que el funcionamiento es inestable y los LEDs parpadean constantemente. Esto ocurre debido a la variabilidad de la señal de sonido.

Para solucionar esto mostramos el siguiente ejemplo.

🛠️ EJEMPLO: Vúmetro con media. Tratamiento de la señal

Como hemos visto la señal de sonido se caracteriza por tener mucha variabilidad. Por ello, resulta conveniente aplicar un tratamiento o filtrado a la señal.

Una técnica efectiva para reducir esta variabilidad es el cálculo de la media móvil (o promedio). Esto implica tomar un número fijo de mediciones sucesivas y promediarlas.

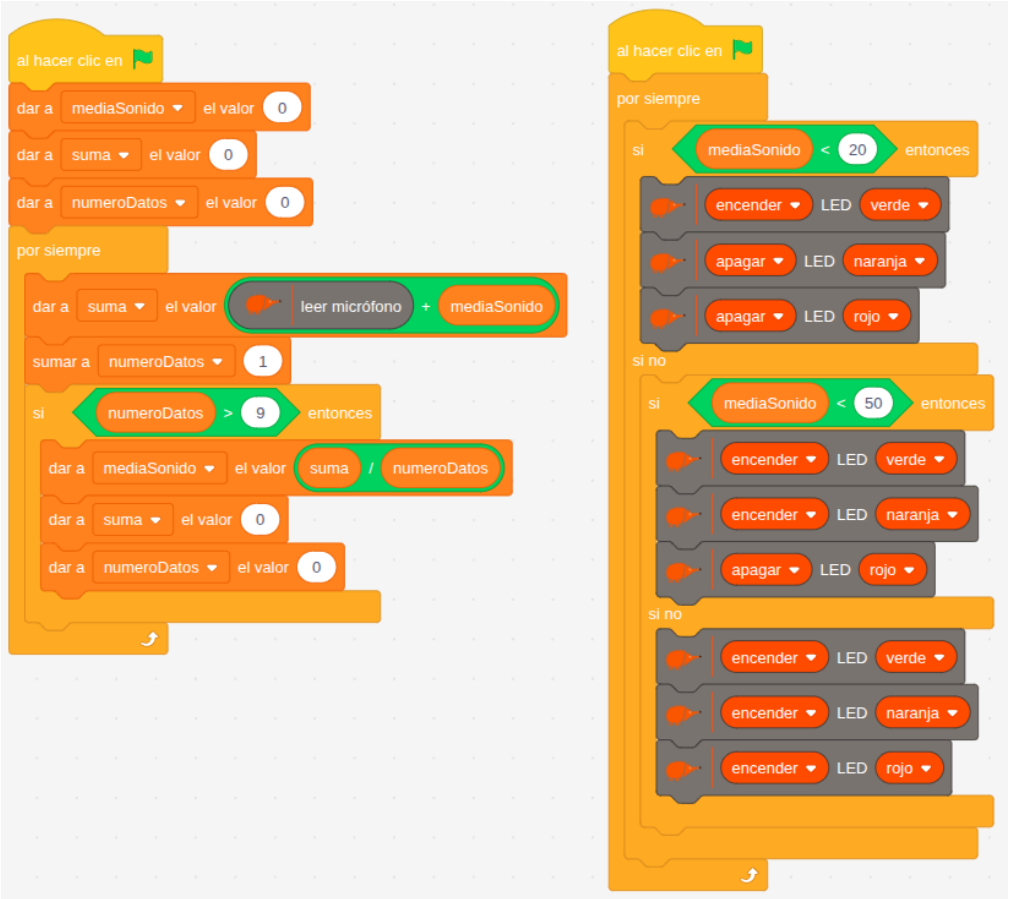
Cálculo de la media:

Para implementar el cálculo de la media:

1. Creamos una variable llamada *suma* para acumular los valores medidos en un ciclo.

2. Cada diez medidas (o el número de muestras predefinido:
    a. Calculamos el valor promedio (*mediaSonido*) dividiendo la suma entre el número de muestras.
    b. Inicializamos las variables *suma* y *númeroDatos*.

De esta forma, se suaviza la lectura y se obtienen valores más estables y representativos de la intensidad del sonido ambiente.



5.1.10 Entrada MKMk: piano

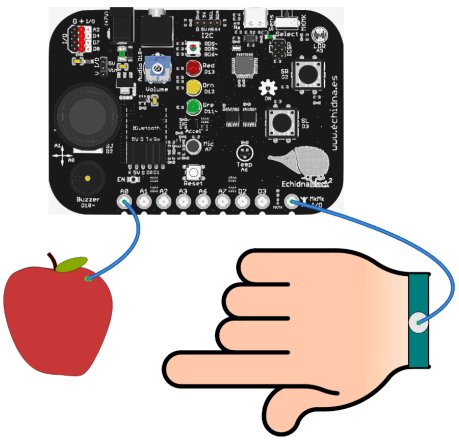
⚠️ **¡RECUERDA!** Para que funcione el modo MkMk debemos poner el **selector** del modo de funcionamiento hacia la derecha, y se nos encenderá el LED testigo en la parte inferior.

⚙️ **COMPONENTE:**

Echidna dispone de 8 conexiones MkMk.

Una conexión MkMk es un conector que permite detectar gran variedad de objetos al conectarlos entre una entrada y el común (El logo Echidna también se comporta como común).

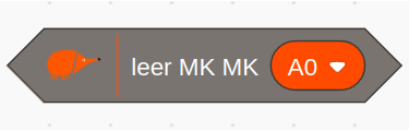
Para detectar elementos debemos conectar un cable al común y otro a una de las entradas.



Entradas MkMk: A0, A1, A2, A3, A6, A7, D2, D3.

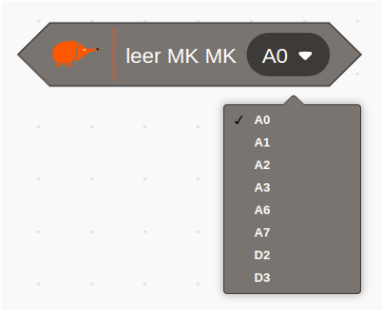
🖥️ BLOQUE DE PROGRAMACIÓN:

Echidna dispone de un bloque de programación específico para las entradas MkMk que nos devuelve un True o un False en función de si detecta que el circuito se ha cerrado o es un circuito abierto.

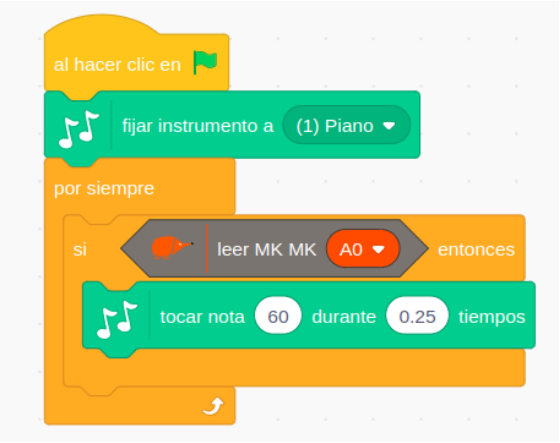


El umbral para cambiar de False a True está establecido en 350 dentro del rango de lecturas analógicas que puede registrar la entrada MKMK (0–1023).

En el bloque podemos seleccionar la entrada MkMk que queramos utilizar:



🛠️ EJEMPLO:



En este ejemplo vamos a ver cómo programar una nota de piano, que suena cuando tocamos la entrada MkMk A0.

En este caso sonará la nota 60 del piano durante 0,25s cada vez que se active la entrada MkMk A0.

Ajustar la sensibilidad de MkMk:

Para ajustar el umbral de activación de las entradas analógicas del módulo MkMk, podemos utilizar el bloque "leer entrada analógica A0".

En este caso el piano suena cuando el valor de la lectura es inferior a 150.

Las entradas analógicas (A0, A1, A2, A3, A6, y A7) permiten definir un límite conductivo específico. Sin embargo, las entradas D2 y D3

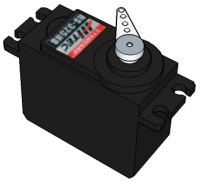
son digitales. Por ello, solo detectan dos estados y no es posible realizar un ajuste del límite conductivo o umbral de activación.

5.2 Componentes complementarios

En este apartado vamos a ver algunos de los componentes que podemos conectar a las entradas/ salidas (I/O) de EchidnaBlack. Estos componentes No vienen incluidos con la placa.

5.2.1 Servomotor de posición: Control posición servo con joystick

⚙️ COMPONENTE:

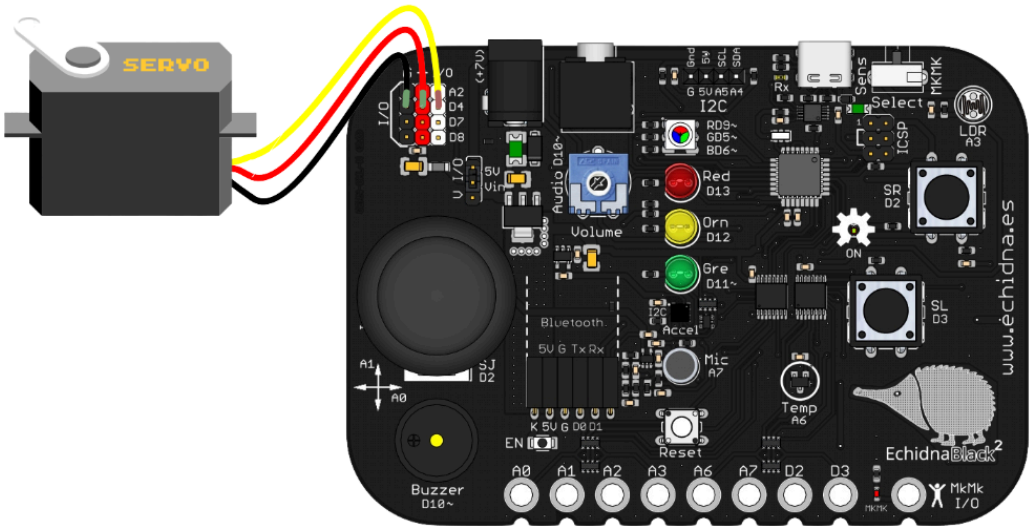


Son motores de corriente continua con una reductora y electrónica de control que permiten posicionarlo en un ángulo entre 0 y 180º.

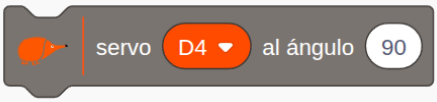
Para conectarlo usamos los pines I/O de entrada-salida (D4, D7, D8, A2).

⚠️ Presta atención al conectar los cables: Vcc, GND y señal, que están indicados por los colores rojo, negro y amarillo.

En caso de que vayas a conectar varios servomotores usa alimentación externa y coloca el selector de alimentación en la posición Vin. Ver apartado 2.4.



🖥️ BLOQUE DE PROGRAMACIÓN:



Para controlar el servo de posicipon podemos usar el siguiente bloque:

En el bloque podemos seleccionar el pin al que conectamos nuestro servo y el ángulo de giro.

Pines: podemos seleccionar los pines; D4, D7, D8 y A2.

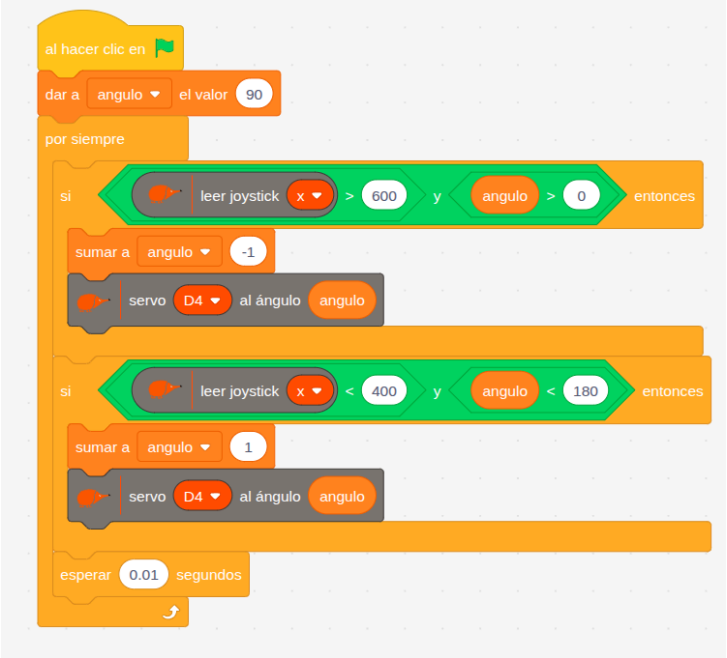
Ángulo de giro: 0-180º.

🛠️ **EJEMPLO:** Control posición servo con *joystick*

Este ejemplo muestra cómo controlar la posición angular de un servomotor utilizando el Eje X del joystick como entrada. La variable *ángulo* almacena la posición deseada y es la que determina la posición del motor.

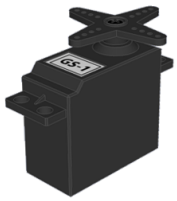
El programa ajusta el valor de la variable *ángulo* mediante dos condiciones:

- Si el valor del joystick x es mayor de 600 y el ángulo menor de 180º incrementamos en uno el valor de la variable *ángulo* y ajustamos la posición del servo.
- Si el valor del joystick x es menor de 600 y el ángulo mayor de 0º decrementamos en uno el valor de la variable *ángulo* y ajustamos la posición del servo.



5.2.2 Servomotor de rotación continua: Control de sentido de giro con pulsadores

⚙️ **COMPONENTE:**



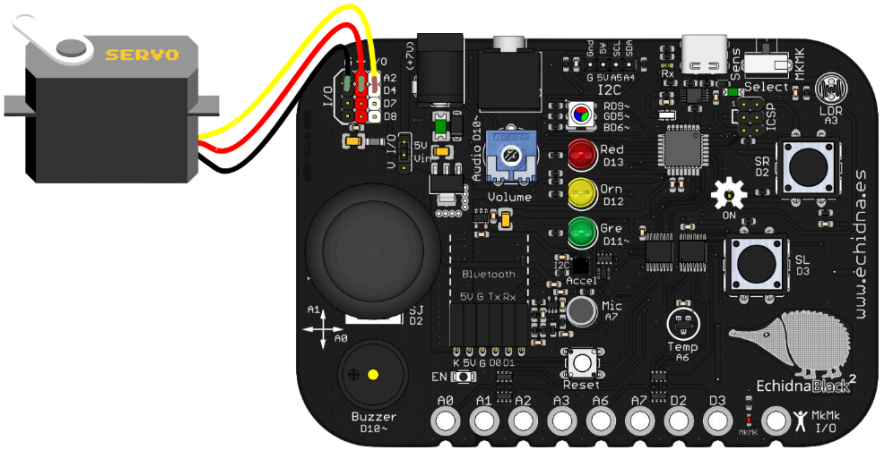
Son motores de corriente continua con una reductora y electrónica de control que permiten controlar el sentido de giro.

Permiten una pequeña variación de velocidad.

Para conectarlo usamos los pines I/O de entrada-salida (D4, D7, D8, A2).

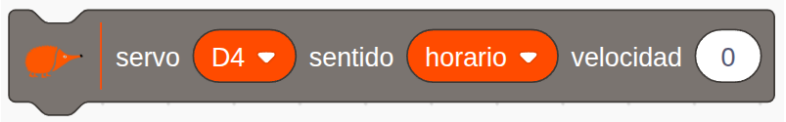
⚠️ Presta atención al conectar los cables: Vcc, GND y señal, que están indicados por los colores rojo, negro y amarillo.

En caso de que vayas a conectar varios servomotores usa alimentación externa y coloca el selector de alimentación en la posición Vin. Ver apartado 2.4.



🖥️ BLOQUE DE PROGRAMACIÓN:

Para controlar el servo de posición podemos usar el siguiente bloque:



En el bloque podemos seleccionar el pin al que conectamos nuestro servo y el sentido de giro y la velocidad.

Pines: podemos seleccionar los pines; D4, D7, D8 y A2.

Sentido de giro: horario/ antihorario

Velocidad: 0-100%

🛠️ EJEMPLO: Control de sentido de giro con pulsadores

Este ejemplo ilustra cómo controlar el sentido de giro de un servomotor continuo (360 grados) utilizando dos pulsadores como entradas.

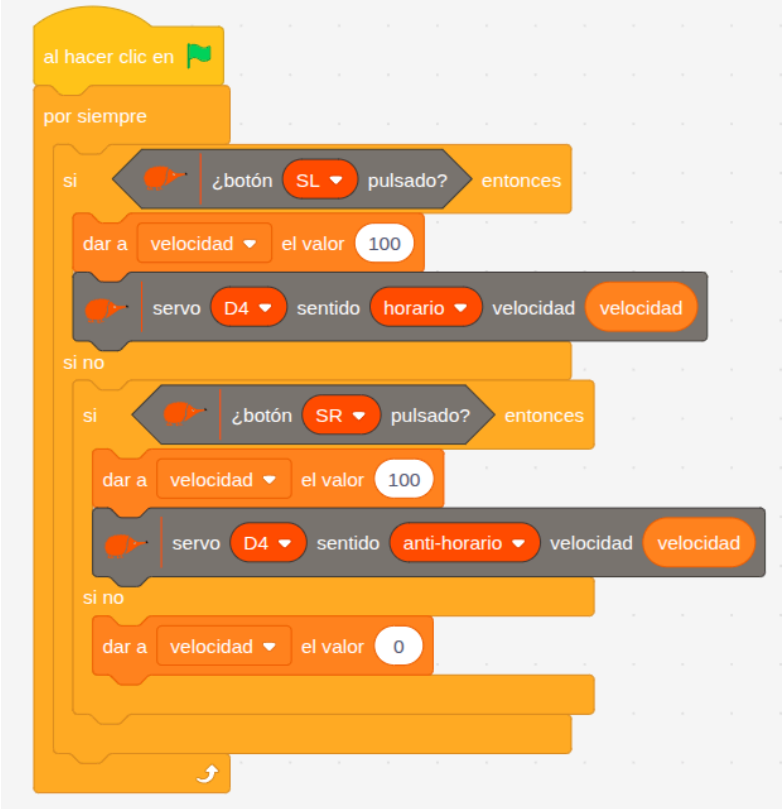
Lógica de Funcionamiento:

El programa evalúa el estado de los pulsadores para determinar la dirección del giro o la detención del servomotor. Utilizamos la variable velocidad para controlar encendido/ apagado del motor.

Giro Horario: SI el pulsador SL (Izquierdo) está presionado, el servomotor gira en sentido horario.

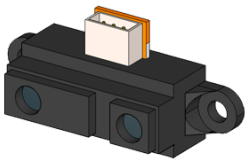
Giro Antihorario: SI NO, SI el pulsador SR (Derecho) está presionado, el servomotor gira en sentido antihorario.

Detención: SI NO (es decir, si ninguno de los dos pulsadores está presionado), el servomotor se detiene.



5.2.3 Sensor de distancia de infrarrojos: Sistema de asistencia al estacionamiento

⚙️ **COMPONENTE:**



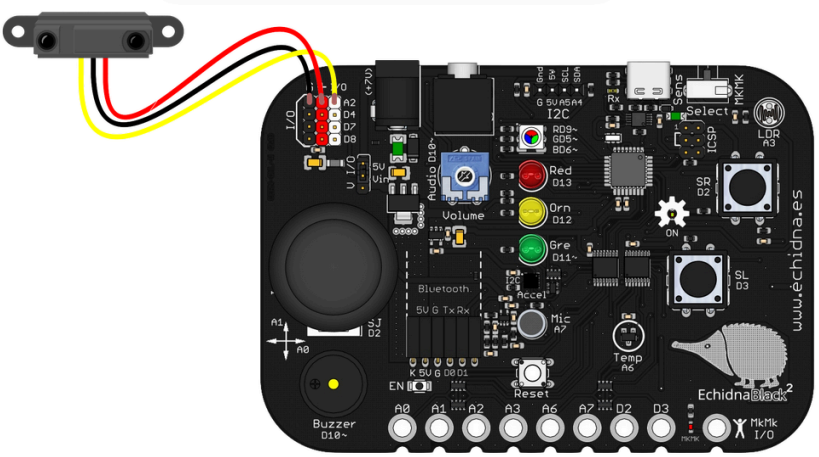
Es un sensor de distancia que proporciona una tensión según la cantidad de infrarrojo que rebota en una superficie.

Salida Inversamente Proporcional y No Lineal: El valor de salida es inversamente proporcional a la distancia al objeto. Es decir, el valor aumenta cuanto más se acerca el objeto. Además, la relación entre el valor de salida y la distancia no es lineal, por lo que requiere calibración o el uso de tablas para obtener una distancia precisa.

Rango de medida: El rango efectivo de medición varía significativamente según el modelo específico del sensor utilizado.

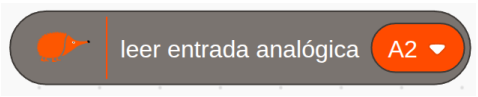
Susceptibilidad a la luz: La precisión del sensor puede verse afectada por la presencia de luz ambiente intensa o fuentes de infrarrojos externas.

Conexión: Se conecta directamente a 5v, GND y A2.



🖥️ BLOQUE DE PROGRAMACIÓN:

Utilizaremos el bloque genérico, leer entrada analógica, seleccionando la entrada A2.



Valores: el rango de valores de la entrada analógica es 0-1023.

🛠️ EJEMPLO: Sistema de asistencia al estacionamiento

Este ejemplo simula un sistema de ayuda al estacionamiento.

El sensor de distancia IR conectado en el pin de entrada/salida (I/O A2) mide continuamente la distancia a un objeto. La funcionalidad del programa es modular la frecuencia del zumbador en función de esta distancia.

1º Medir valores distancia:

Lo primero es medir los valores de distancia y almacenarlos en la variable *lecturaIR*, que se utiliza para monitorizar y mostrar el valor del sensor.

Para ello podemos usar este programa:

Al hacer clic en la bandera verde, se lee continuamente el valor del sensor conectado al pin analógico A2 y se guarda en la variable

lecturaIR. Cada medida se realiza con un intervalo de 0,2 segundos, para estabilizar la lectura.

Puedes activar la casilla de verificación de la variable para ver el valor registrado.

2º selección de umbrales:

El siguiente paso será seleccionar tres umbrales diferentes poniendo un obstáculo frente al sensor a las tres distancias que queramos definir. En este caso usamos estos tres valores:

- Distancia Pequeña: 200
- Distancia Muy Pequeña: 400
- Distancia Mínima: 600

3º Programa

Tenemos dos hilos de ejecución:

Hilo 1 de adquisición de datos:

Medimos el valor del sensor y lo almacenamos en la variable *lecturaIR*.

Hilo 2 Lógica de programación:

Zona Segura: Primero se comprueba si la distancia es menor de 200; en ese caso, el zumbador permanece apagado.

Zona de Advertencia: Si no, se evalúa si la variable *lecturaIR* es menor de 400 (200≤distancia<400). Cuando esto sucede, los pitidos se encienden y apagan con pausas de 0,2 segundos.

Zona de Peligro Alto: Si no, se comprueba si la variable *lecturaIR* es menor de 600 (400≤distancia<600). En este rango, el zumbador se enciende y apaga cada 0,1 segundos, produciendo un ritmo más rápido.

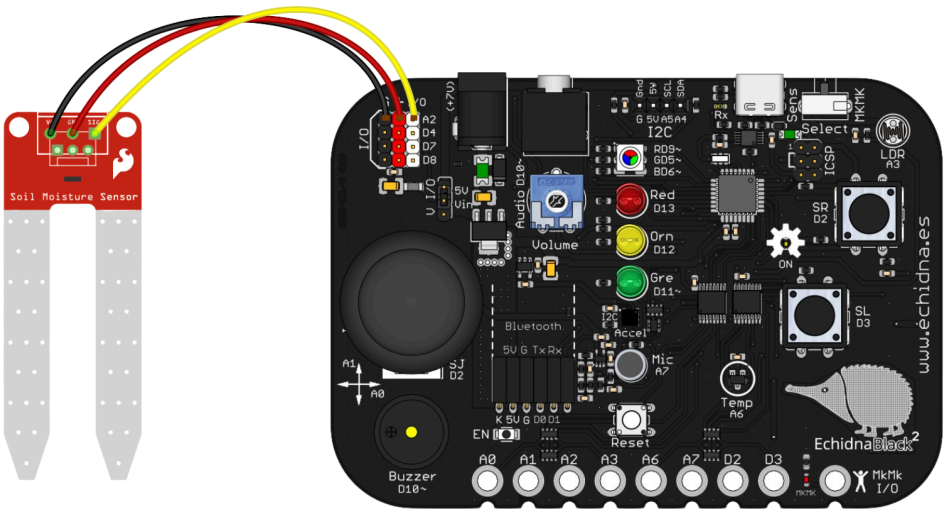
Zona Crítica (Colisión Inminente): Finalmente, si ninguna de las condiciones anteriores se da (es decir, la variable *lecturaIR* es mayor o igual a 600), el zumbador emite pitidos muy rápidos, encendiéndose y apagándose cada 0,05 segundos.

Todo este proceso se repite mediante un bucle por siempre, ajustando el sonido según lo cerca o lejos que esté el objeto.

5.2.4 Sensor de humedad del suelo: Monitorización de riego

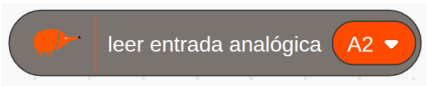
⚙️ COMPONENTE:

El sensor de humedad del suelo permite medir la cantidad de agua en la tierra. Tiene dos componentes principales: los electrodos, que se introducen en la tierra, y un circuito electrónico que convierte la humedad detectada en una señal eléctrica. Cuando el suelo contiene más agua, la conductividad entre los electrodos aumenta y, por tanto, el sensor devuelve un voltaje mayor; cuando el suelo está seco, la conductividad disminuye y el voltaje es menor.



🖥️ BLOQUE DE PROGRAMACIÓN:

Utilizaremos el bloque genérico, leer entrada analógica, seleccionando la entrada A2.



Valores: el rango de valores de la entrada analógica es 0-1023.

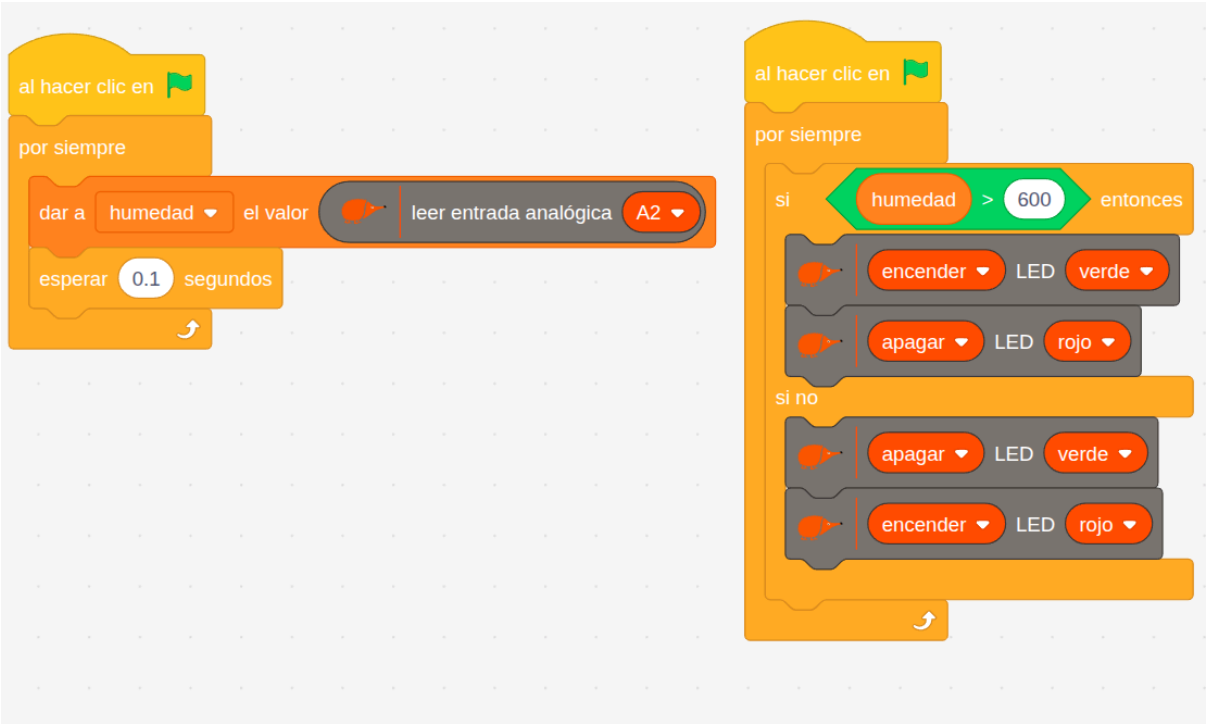
🛠️ EJEMPLO: Monitorización de riego

En el ejemplo vemos cómo hacer un sistema que nos indique cuando hay que regar la planta.

- Si la humedad es adecuada lo indicamos con el LED verde.
- Si la humedad es baja y la planta necesita ser regada lo indicamos con el LED rojo.

Lo primero es leer el sensor de humedad y ver y almacenar los valores que proporciona.

En función de estos valores encenderemos el LED que corresponda a la humedad registrada.



6. LEARNINGML

6.1 Introducción a LearningML

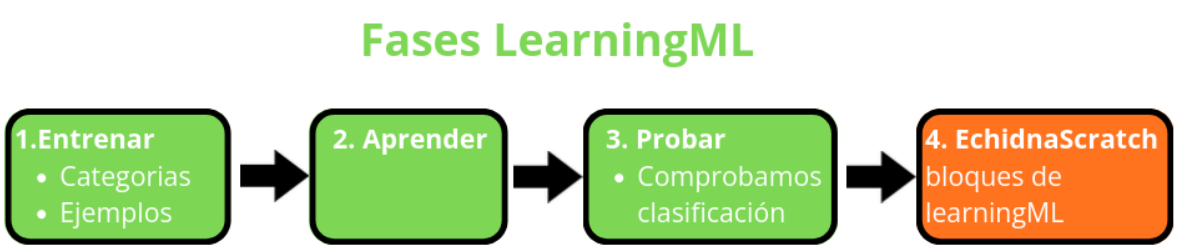
LearningML es una herramienta educativa diseñada para que el alumnado aprenda los conceptos básicos del machine learning de forma sencilla, visual y manipulativa. El alumnado puede crear modelos que clasifican imágenes, textos o números y después utilizarlos dentro de proyectos con bloques de Scratch.

Su principal objetivo es que el alumnado comprenda cómo se entrenan los modelos, qué datos necesitan, cómo influyen los sesgos y cómo se usan después para tomar decisiones. Todo ello sin programación compleja, favoreciendo el pensamiento computacional y el análisis crítico de la inteligencia artificial.

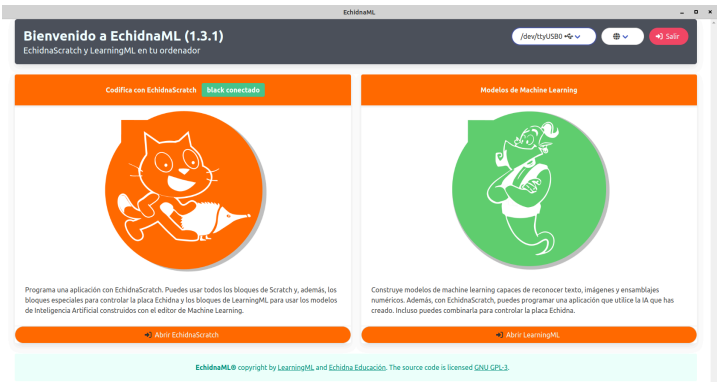
EchidnaML incluye esta herramienta, que puede usarse por sí misma o en combinación con los bloques de robótica con EchidnaScratch.

Fases para crear un modelo LearningML y

1. Creamos las categorías y los ejemplos.
2. Entrenamos el modelo.
3. Comprobamos que el modelo clasifica correctamente.
4. Abrimos EchidnaScratch y usamos los bloques de LearningML.



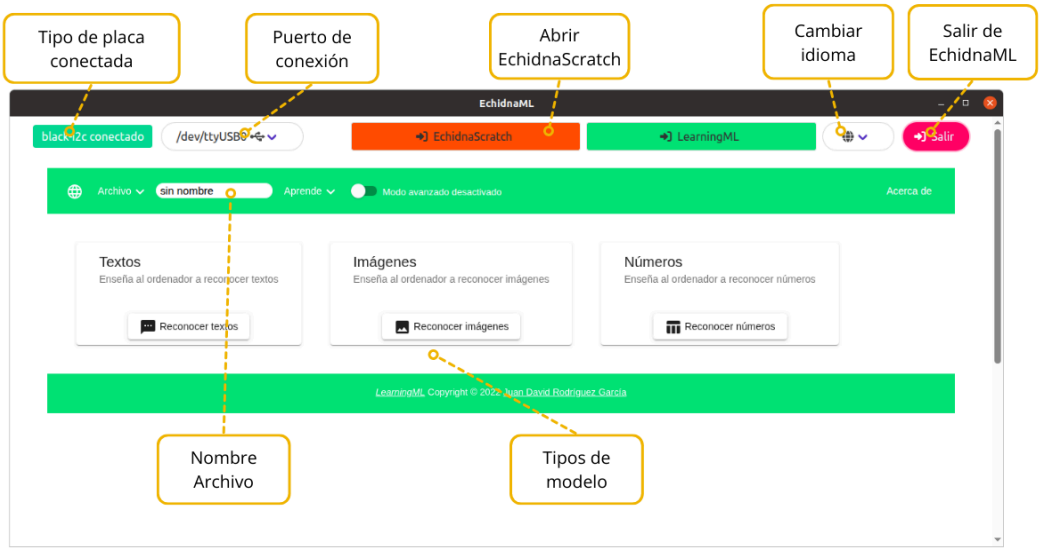
Acceder a LearningML:



Para acceder a learningML lo podemos hacer a través de la pantalla de inicio de EchidnaML en el icono.

Al acceder a la pantalla de LearningML podemos:

- Elegir el tipo de modelo que queremos crear: texto, imagen o números.
- Seleccionar el idioma.
- Ver la placa y el puerto de conexión al que se ha conectado, en caso de que hayamos conectado alguna.
- Abrir EchidnaScratch.
- Salir del programa.

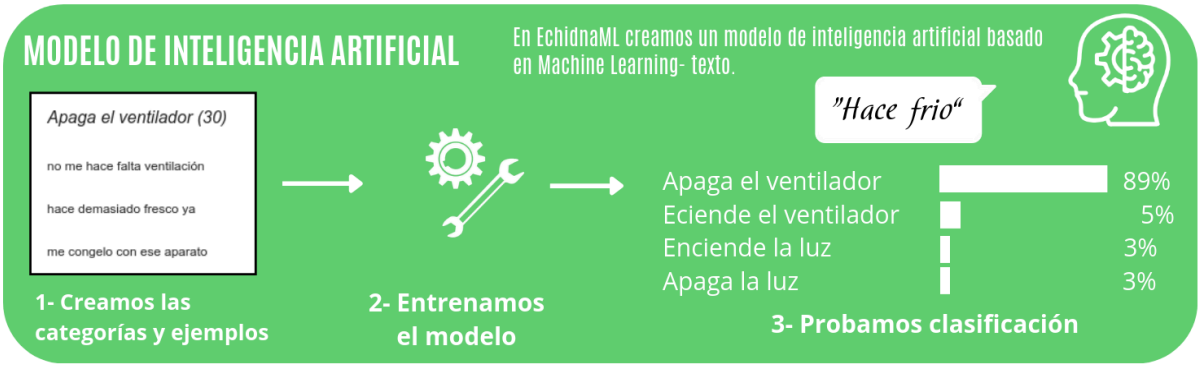


A continuación, encontrará instrucciones para crear modelos de reconocimiento de texto, imagen y números. Para ampliar la información sobre su uso, puede consultar la web del proyecto.

6.2 Crear un modelo de texto

Vamos a ver los pasos para crear un **modelo de texto** en LearningML y como usarlo con EchidnaScratch.

En este caso crearemos un asistente virtual que nos controla la iluminación de la vivienda y el ventilador.



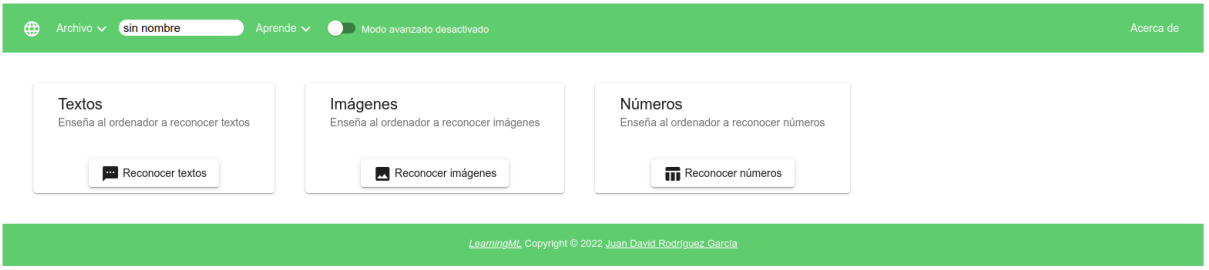
A. Abrir la aplicación Modelos de Machine Learning

Una vez hemos abierto EchidnaML abrimos la aplicación Modelos de Machine Learning (LearningML).

B. Elegir tipo de datos.

Lo primero que tenemos que hacer es elegir con qué tipo de datos vamos a trabajar, podemos hacerlo con textos, imágenes y números.

En nuestro caso vamos a trabajar con datos de tipo texto.

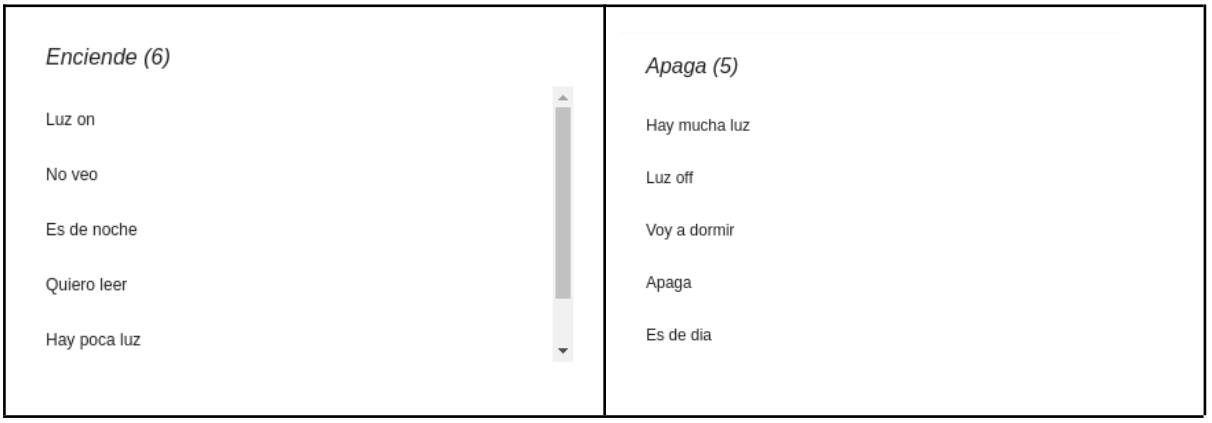


C. Entrenar: añadir las clases y los ejemplos

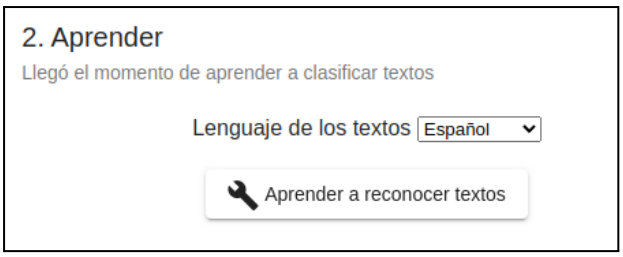
Una vez que hemos elegido el tipo de datos creamos las clases y proporcionamos los datos (textos) para que el algoritmo aprenda a reconocerlas.

En nuestro caso vamos a crear dos clases:

Enciende y *Apaga*. Estas nos van a permitir controlar el encendido del LED. Cuanto más textos se añadan, mejor será el modelo construido, es decir, acertará más cuando clasifica nuevos textos.

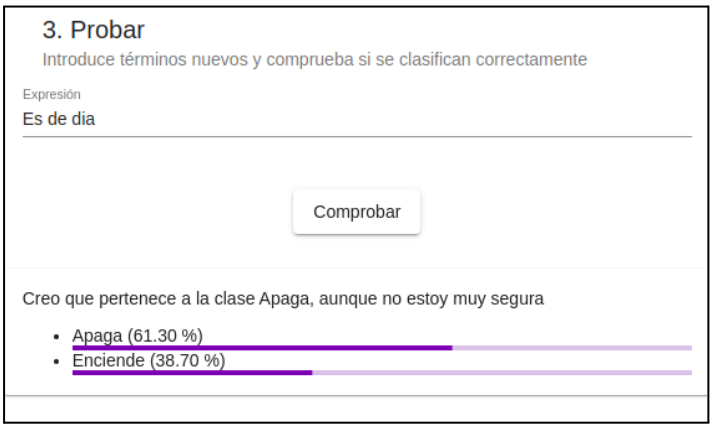


D. 2-Aprender



Primero pulsamos el botón *2. Aprender* a reconocer textos. Entonces el algoritmo de machine learning, analizando los ejemplos que hemos introducido, construye un modelo de Inteligencia Artificial que es capaz de reconocer nuevos textos. Es lo que se conoce como aprendizaje a partir de datos, y es la base del machine learning.

E. 3-Probar.



Una vez construido el modelo, debemos realizar pruebas para comprobar si el modelo clasifica correctamente y con que % de confianza lo hace. En esta fase debemos usar textos distintos, aunque similares, a los que introdujimos durante el entrenamiento.

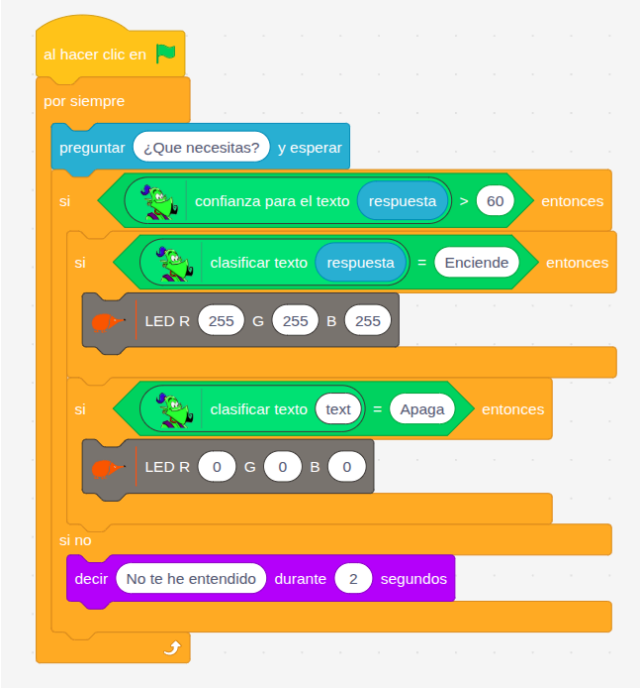
En este caso comprobamos que clasifica la frase “Es de día” con un 61% de probabilidad en la categoría Apaga. Lo cual es correcto.

F. 1-volver a entrenar?

Si el modelo no funciona como queremos o queremos mejorarlo hay que añadir más datos y revisar los de la fase de entrenamiento. Ten en cuenta que, cuanto más datos (textos), si están bien elegidos, mejor será la calidad del modelo, es decir, más aciertos producirá el modelo cuando clasifique.

G. Programamos en EchidnaScratch

Una vez que las pruebas de entrenamiento hayan sido satisfactorias, abrimos Echidna Scratch. A continuación, podremos combinar los bloques de Machine Learning con el modelo que acabamos de generar y programar nuestra aplicación.



Este ejemplo muestra a un modelo de Machine Learning usado para clasificar una entrada de texto y, con el resultado, tomar una decisión en la placa robótica: encender o apagar el LED RGB.

Consulta y Clasificación: El personaje pregunta qué necesitamos y envía la respuesta que introduce el usuario al modelo de machine learning para su clasificación.

Verificación de Confianza: El sistema comprueba el índice de confianza que el modelo asigna a su clasificación. La acción sólo se ejecuta SI este índice es superior al 60%.

En caso de que la Confianza sea menor del 60% el programa dice “No te he entendido”.

Ejecución de Comando: Si la confianza es suficiente, el sistema determina la clase:

SI la clase es "Enciende": El programa envía la instrucción para encender el LED.

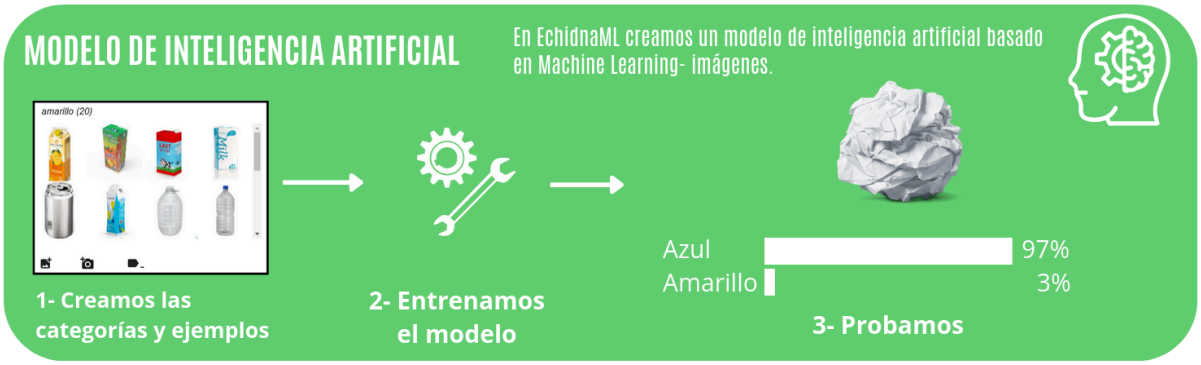
SI la clase es "Apaga": El programa envía la instrucción para apagar el LED.

Al finalizar, el personaje vuelve a decir: “¿Qué necesitas?”, indicando que está listo para una nueva clasificación.

6.3 Crear un modelo de imágenes

En este segundo ejemplo veremos los pasos para crear un **modelo de imagen** en LearningML y como usarlo con EchidnaScratch.

Vamos a crear un modelo que nos clasifique los envases y papeles en las categorías amarillo y azul.

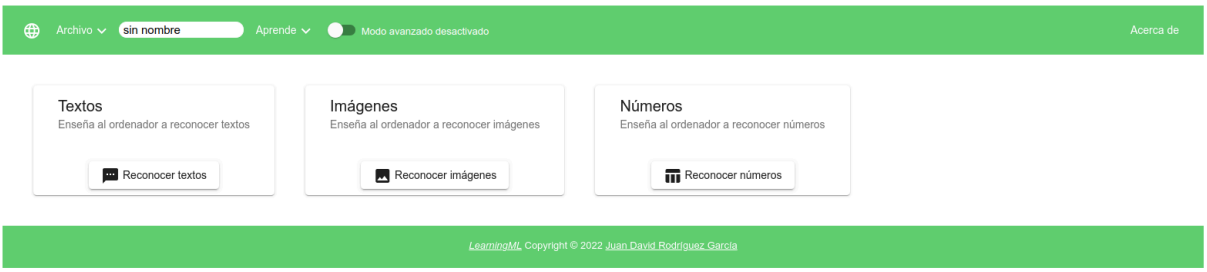


A. Abrir la aplicación Modelos de Machine Learning

Una vez hemos abierto EchidnaML abrimo la aplicación Modelos de Machine Learning.

B. Elegir tipo de datos.

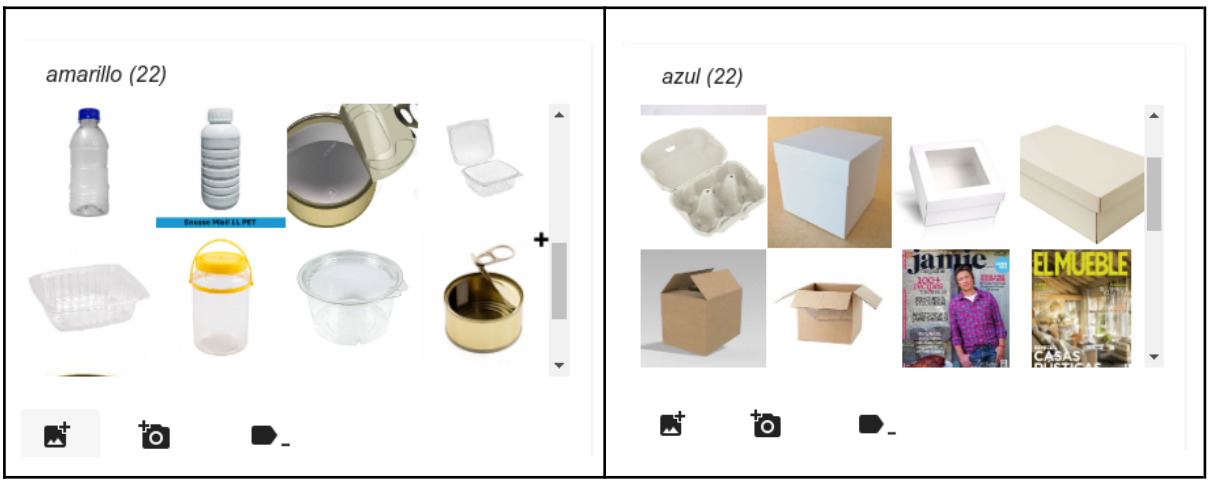
Es el momento de elegir con qué tipo de datos vamos a trabajar, en este caso con datos de imágenes.



C. Entrenar: añadir las clases y los ejemplos

Una vez hemos elegido el tipo de datos creamos las clases y le proporcionamos datos para que el algoritmo aprenda a reconocerlas.

Crearemos dos clases: **Amarillo** y **Azul**, que nos permitirá reconocer residuos que se reciclan en el contenedor azul y residuos que se reciclan en el contenedor amarillo, para controlar dos servomores que abran el contenedor adecuado.



D. 2-Aprender

Pulsamos el botón *2. Aprender a reconocer imágenes*, para que el algoritmo de machine learning analice las imágenes y construya un modelo de Inteligencia Artificial capaz de reconocer imágenes similares pero distintas a las que hemos usado durante el entrenamiento.

La construcción del modelo, que se denomina aprendizaje, puede tardar varios segundos. Si las imágenes de ejemplo de la fase de entrenamiento son muchas, este proceso puede tardar bastante. Esto es una característica de los algoritmos de machine learning, requieren mucho tiempo de proceso y bastante memoria.

E. 3-Probar

Cuando el algoritmo finaliza ya tenemos disponible el modelo de machine learning. Es el momento de probar cómo de bien funciona.A

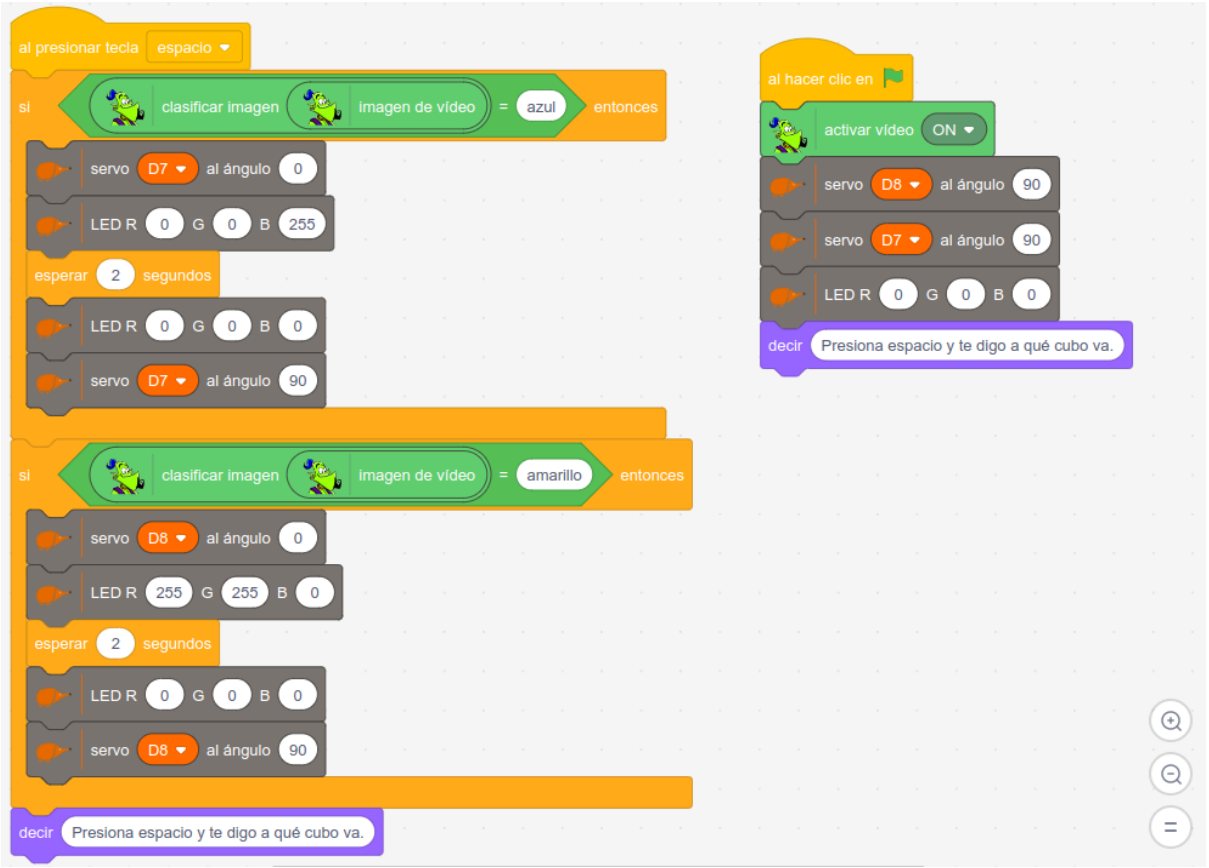
Añadimos imágenes nuevas, no usadas en el entrenamiento y hacemos pruebas para comprobar si el modelo clasifica correctamente y con qué porcentaje de confianza lo hace.

En el ejemplo que mostramos al lado, comprobamos que clasifica la imagen con más de un 88% de probabilidad en la categoría amarillo, lo cual es correcto.

F. 1-volver a entrenar?

Si no funciona como queremos o deseamos mejorarlo hay que añadir y revisar los datos de la fase de entrenamiento.

G. Programamos en EchidnaScratch



Una vez que las pruebas de entrenamiento hayan sido satisfactorias, podemos acceder a Echidna Scratch. Desde allí, ya podremos utilizar los bloques de machine learning con el modelo que acabamos de generar y programar nuestra aplicación robótica.

Este ejemplo implementa un sistema automatizado que utiliza un modelo de Machine Learning para clasificar una imagen capturada por la cámara y, en función del resultado, activa los actuadores del robot (servos y LED RGB)

Configuración Inicial:

Al inicio del programa, el sistema se prepara para recibir la clasificación:

- Se activa la cámara para la captura de vídeo.
- Los servos D7 y D8 se colocan en la posición de 90° (posición de cerrado o reposo).
- El LED RGB se asegura de estar apagado.

El personaje indica:

"Presiona espacio y te digo a qué cubo va", señalando que el sistema está listo para clasificar.

Lógica de Clasificación (Al presionar Espacio)

Al presionar la tecla Espacio, el programa clasifica el fotograma de vídeo capturado y ejecuta la acción robótica correspondiente:

SI clasifica en Categoría Azul**:**

- El servo D7 se mueve a 0° (abre el cubo azul).
- El LED se ilumina en AZUL (RGB 0, 0, 255).

SI clasifica en Categoría Amarilla:

- El servo D8 se mueve a 0° (abre el cubo amarillo).
- El LED se ilumina en AMARILLO (RGB 255, 255, 0)

Al finalizar, el personaje vuelve a decir: “Presiona espacio y te digo a qué cubo va”, indicando que está listo para una nueva clasificación.

6.4 Crear un modelo de números

En este tercer ejemplo con LearningML veremos los pasos para crear un **modelo de números** y como usarlo con Echidna Scratch.

En este caso vamos a crear un **modelo** que nos clasifique la inclinación de la placa detectando: **derecha** e **izquierda**. En Echidna Scratch vamos a desplazar el personaje en la dirección que nos clasifique.

A. Abrir la aplicación Modelos de Machine Learning

Una vez hemos abierto EchidnaML abrimo la aplicación Modelos de Machine Learning.

B. Elegir tipo de datos.

Es el momento de elegir con qué tipo de datos vamos a trabajar, en este caso con datos de tipo **números**.

C. Entrenar: añadir las clases y los ejemplos

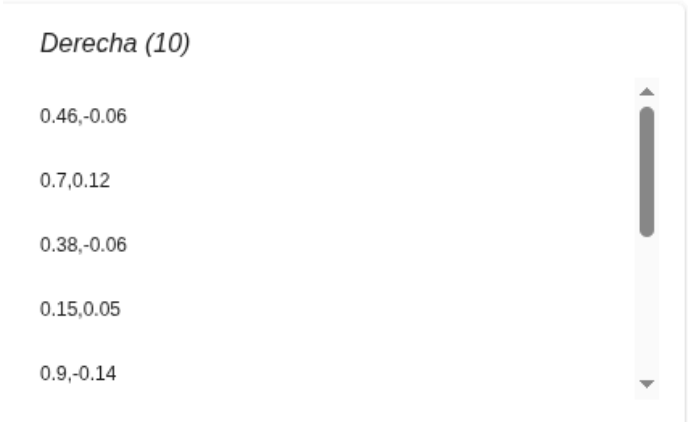
Una vez hemos elegido el tipo de datos números, seleccionamos el número de columnas del número. En nuestro caso elegimos números con 2 columnas, ya que vamos a introducir datos de los valores del acelerómetro en coordenadas x e y.

Creamos las clases y le proporcionamos datos para que el algoritmo aprenda a reconocerlas.

En este caso vamos a crear dos clases:

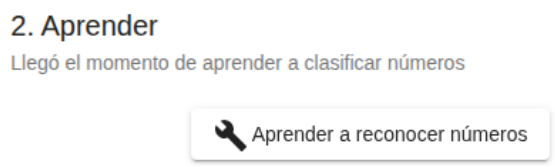
- Derecha
- Izquierda

Los números se deben introducir separándolos con una coma. Cómo hemos elegido un número de columnas igual a 2, debemos introducir un par de números separados por una coma.



D. 2-Aprender

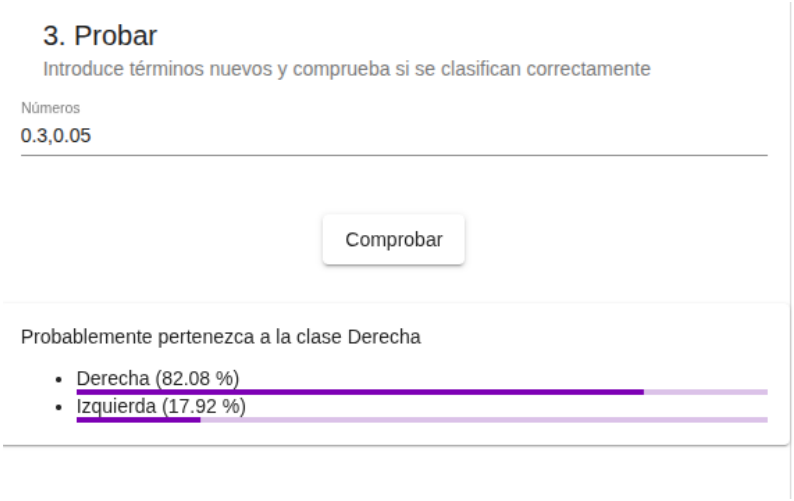
Al hacer clic en el botón *Aprender a reconocer números*, como ya hemos visto en los otros tipos, el algoritmo de machine learning aprende a reconocer números a partir de nuestros datos.



E. 3-Probar

Es el momento de probar que el modelo que hemos creado funciona correctamente. Para lo cual introducimos en la caja de texto de la fase 3-Probar, nuevos números separados por coma, similares pero distintos a los de la fase de entrenamiento.

El modelo arrojará su predicción y comprobaremos si clasifica correctamente y con qué porcentaje de confianza lo hace.



En este caso comprobamos que clasifica números de diferentes sectores con más de un 70% de probabilidad en la categoría que le corresponde.

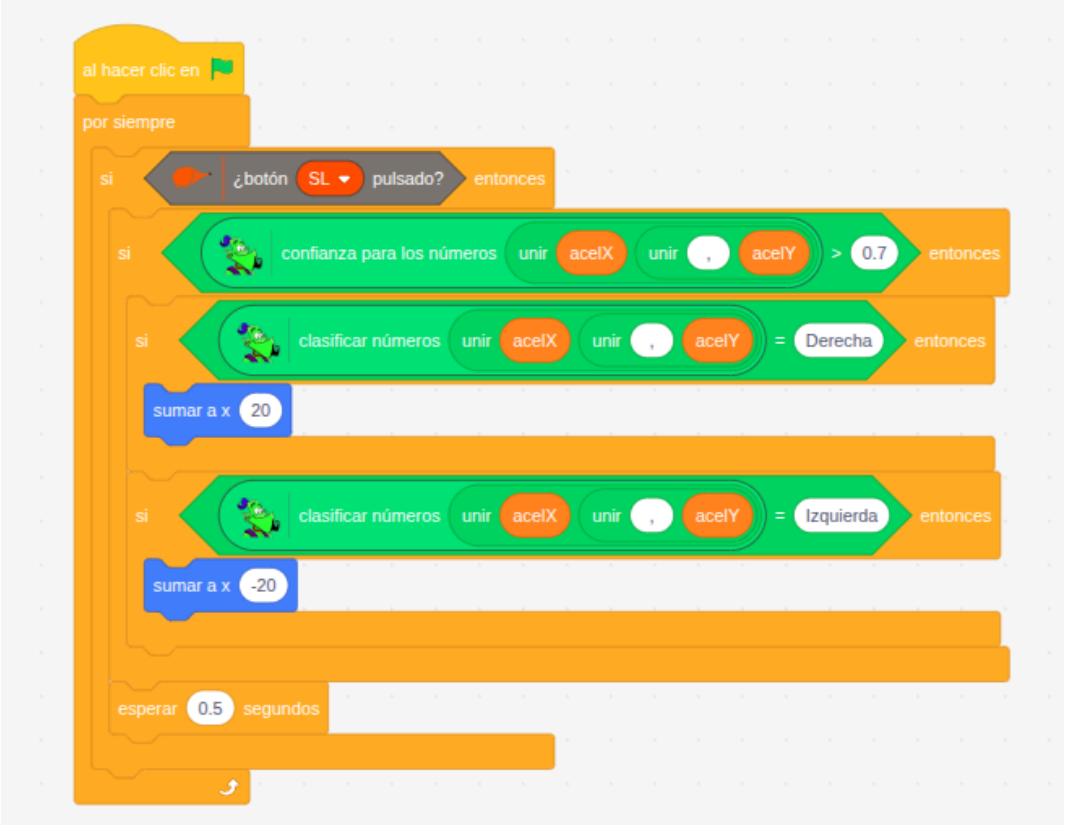
F. 1-volver a entrenar?

Si no funciona como queremos habrá que añadir y revisar los datos de la fase de entrenamiento,

G. Programamos en EchidnaScratch

Una vez que las pruebas del modelo hayan sido satisfactorias, podemos acceder a Echidna Scratch. Desde allí, ya podremos utilizar los bloques de Machine Learning con el modelo que acabamos de generar y programar nuestra aplicación robótica.

En este ejemplo inclinamos la placa en diferentes posiciones, cuando le damos al botón SL desplaza el personaje en dirección Derecha o izquierda..

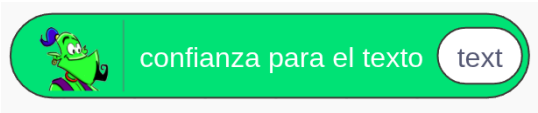


6.5 Bloques LearningML en EchidnaScratch

Bloques reconocer textos:

Clasifica Texto: Es un bloque de tipo “reporter”. Devuelve la etiqueta clasificada como más probable del texto que se introduzca como argumento.

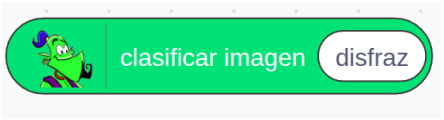
Confianza para texto: Es un bloque de tipo “reporter”. Devuelve la probabilidad en porcentaje (0-100) de la clasificación propuesta por el modelo.



Bloques reconocer imágenes:

Clasificar imagen: Este reporter devuelve el valor de la clasificación dada por el modelo de Machine Learning a la imagen que se aporta como primer argumento. Dicho argumento puede ser:

- El nº de disfraz cuya imagen se quiere clasificar
- El disfraz actual dado por el reporter "disfraz actual"
- La imagen tomada por la webcam y representada por el reporter



Confianza para la imagen: Este reporter devuelve la probabilidad asignada por el modelo a la clasificación más probable (es decir a la que devuelve el reporter anterior) de la imagen que se aporta en su argumento, que igual que antes puede ser:

- El nº de disfraz cuya imagen se quiere clasificar
- El disfraz actual dado por el reporter
- La imagen tomada por la webcam y representada por el reporter

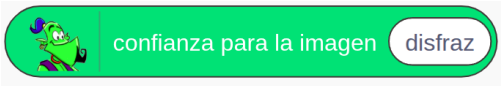
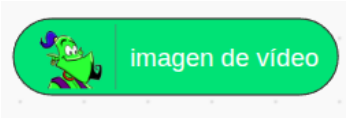
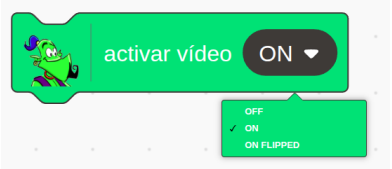


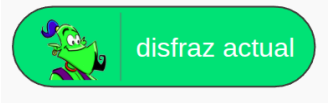
Imagen de vídeo: Un reporter que devuelve la imagen tomada por la webcam



Activar vídeo: Un comando con el que se puede activar la webcam, activar la webcam en modo invertido desactivar la webcam.

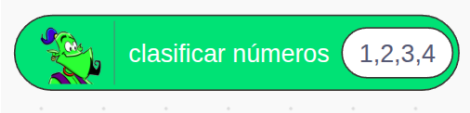


Disfraz actual: Un reporter que devuelve el disfraz actual activo.

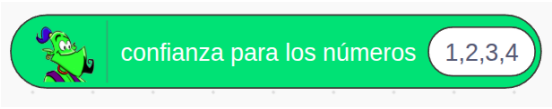


Bloques reconocer números:

Clasificar números: Es un bloque de tipo “reporter”. Devuelve la etiqueta clasificada como más probable del conjunto de números que se introduzca como argumento.



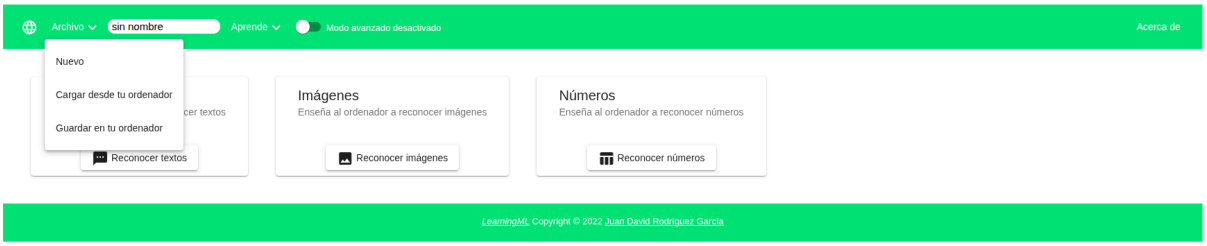
Confianza para los números: Es un bloque de tipo “reporter”. Devuelve la probabilidad en % (0-100) de la clasificación propuesta por el modelo.



6.6 Exportar e importar modelos

La versión de LearningML en EchidnaML es un programa de Escritorio, por lo que si queremos guardar nuestros modelos debemos descargarlos y almacenarlos en nuestro ordenador. Para ello debemos pulsar en *Archivo/ Guardar en tu ordenador* y almacenarlo en la carpeta que elijamos.

Para recuperar el proyecto debemos pulsar en *Archivo/ Cargar desde tu ordenador* y buscar el archivo en la carpeta donde lo almacenamos.



Los modelos se almacenan en formato **json** y son compatibles con los modelos realizados en LearningML escritorio y online.

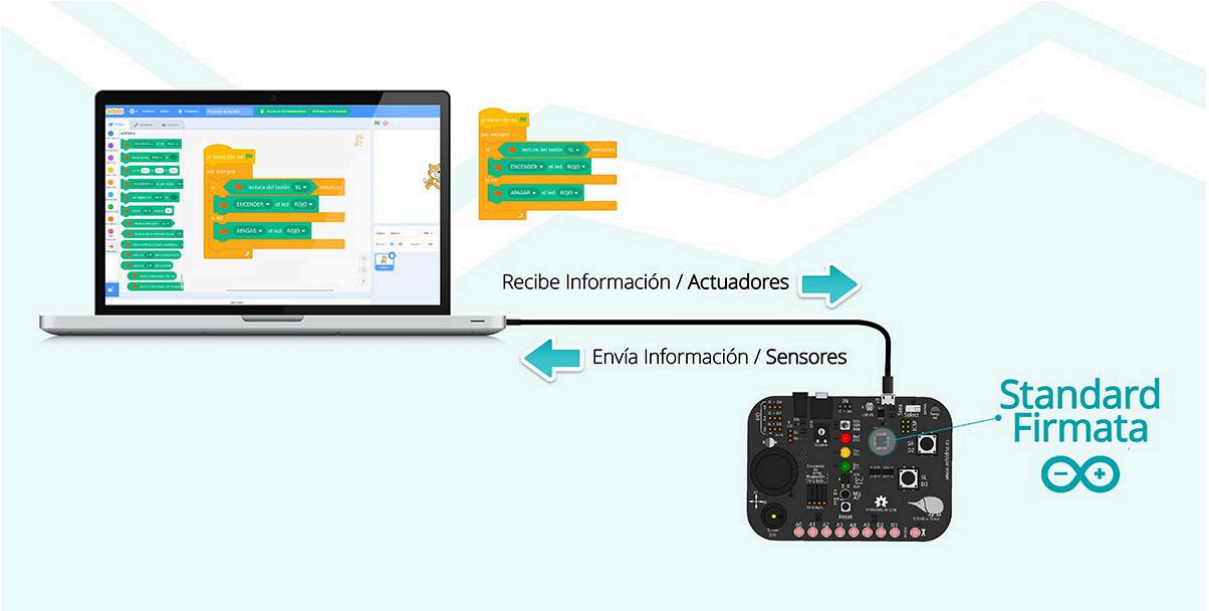
7. PROGRAMA INSTALADO EN EL MICROCONTROLADOR

7.1 ¿Qué es Firmata?

Para trabajar con EchidnaML en la placa EchidnaBlack tenemos que tener instalado un programa denominado **Firmata** que permite la comunicación entre la placa y el ordenador.

Firmata es un protocolo para comunicar microcontroladores con la computadora de una manera sencilla. Permite que se ejecute un programa en EchidnaLM o Snap4Arduino y que este se comunique con la placa microcontroladora mediante el puerto serie.

De modo que la placa envía información sobre el estado de los sensores y el programa que realizamos en EchdinaML lo procesa y devuelve información sobre el estado de los actuadores de la placa.



En EchidnaBlack trabajamos con **StandardFirmata**, que viene **instalado** de **serie** por lo que lo normal es que no necesites hacer nada.

Si necesitas volver a **instalar** el programa **StandardFirmata** en el siguiente apartado te explicamos cómo hacerlo.

7.2 Cómo instalar StandardFirmata

Vamos a ver como instalar StandardFirmata usando el **IDE de Arduino**, pero también puedes usar otros programas como:

- PlatformIO
- Eclipse Arduino IDE

- Codebender
- ArduinoDroid
- Programino

Con el IDE de Arduino debemos seguir los siguientes pasos:

1- Instalar IDE Arduino: El primer paso será tener instalado en nuestro ordenador el IDE de Arduino. Está disponible para Linux, MacOs y Windows y te lo puedes descargar desde la propia página de Arduino, donde también tienes una guía para instalarlo.

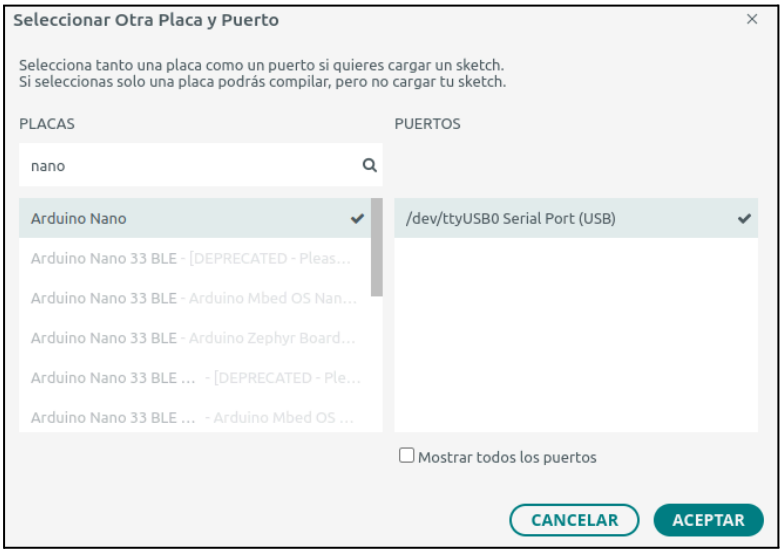
2- **Conectamos** la **placa EchidnaBlack** a nuestro ordenador a través del puerto USB.

3- **Abrimos** el **IDE de Arduino**.

4- **Seleccionamos la placa Arduino** que estemos usando y el **puerto USB** al que se conecta.

Placa Arduino: si tu placa es EchidnaBlack², debes escoger **Arduino Nano**.

En Herramientas → Placa→ Arduino Nano.



Puerto USB: tenemos que seleccionar el puerto USB al que se conecta la placa.

Seguramente aparezca una indicación del puerto USB al que está conectado tu Echidna.

En función del sistema operativo que tengas te aparecerá un **nombre** para el **puerto**:

- GNU Linux: /dev/ttyUSB0
- Windows: COM21
- MACOS: /dev/cu.usbserial-1410 port (USB)

En el que el número asignado puede variar en función de los dispositivos que tengamos conectados.

5- **Seleccionamos el programa** que vamos a cargar, es decir el **StandardFirmata**.

Lo hacemos desde el menú Archivo → Ejemplos→ Firmata→ StandardFirmata.

Atención si no hemos seleccionado la placa en el paso anterior no nos aparece los programas firmata.

6- **Cargamos el programa en la placa**.

Para ello clica en el botón “Subir”, que indica al IDE Arduino que cargue el programa en la placa. Una vez cargado tu Echidna ya está lista para ser programada con EchidnaML. Aunque desconectes la placa y la guardes, el programa StandardFirmata seguirá instalado en la placa. Cuando la vuelvas a conectar a la computadora, se ejecutará dicho programa y será capaz de comunicarse con EchidnaML. Así que la carga del StandardFirmata solo tendrás que hacerla una vez.

8. PREGUNTAS FRECUENTES

8.1 Qué requisitos técnicos necesito

Para trabajar con EchidnaML se requiere un **ordenador** de **escritorio** o **portátil** con prestaciones básicas. En algunas distribuciones de software libre educativo viene instalada por defecto. Este software no es compatible con Chromebooks, dispositivos móviles ni tablets.

Se **recomienda** disponer, como mínimo, de un ordenador con las siguientes **características**: procesador x86_64 o ARM64 de 4 núcleos a 2GHz, 8 GB de memoria RAM y 5 GB de espacio libre en disco para la instalación y ejecución del software.

EchidnaML es **compatible** con los principales sistemas operativos de escritorio:

- Windows: Windows 10 (64 bits) Windows 11 (64 bits)
- macOS (versiones 12 en adelante)
- Linux (distribuciones basadas en Debian, 64 bits)

Se recomienda mantener el sistema operativo actualizado y disponer de permisos de instalación para garantizar el correcto funcionamiento del programa y la comunicación con la placa EchidnaBlack2.

8.2 Cómo instalo EchidnaML en mi ordenador

Para instalar EchidnaML en tu ordenador debes seguir los siguientes pasos en función del SO que tengas.

GNU Linux:

Al descargar el programa nos descarga un archivo echidna_1.4.0_amd64.deb (o la versión que corresponda).

En GNU Linux tienes diversas formas de instalar el SO.

A- Clicar en el archivo

Hacer doble clic en el archivo.

B- Desde el terminal:

1- Nos debemos mover a la carpeta donde tenemos descargado el archivo

cd Descargas

2- Ejecutar la siguiente instrucción:

sudo dpkg -i echidna_1.4.0_amd64.deb (o la versión que corresponda)

macOS:

Al descargar el programa nos descarga un archivo tipo: echidnaML-v1.4.dmg

Debemos dar permiso para que instale un programa externo de una fuente no confiable. Para instalar una aplicación de una fuente no confiable en macOS, tenemos dos opciones:

Método 1: Permitir la instalación manualmente al abrir el archivo

- Busca la aplicación que deseas instalar en el Finder.
- Haz clic derecho sobre el archivo y selecciona Abrir.
- Verás un mensaje de advertencia. Haz clic en Abrir para confirmar la instalación.

Método 2: Anular los ajustes de seguridad

- Ve a Ajustes del Sistema (o Preferencias del Sistema en versiones anteriores de macOS) y haz clic en Privacidad y Seguridad.
- Desplázate hacia abajo hasta la sección de seguridad y verás el mensaje sobre la aplicación bloqueada.
- Haz clic en el botón Abrir igualmente.

Windows:

Después de descargar el programa para Windows “EchidnaML.msi”, hacemos doble “clic” sobre él, daremos permiso para instalar y listo.

8.3 El programa no detecta la placa

Si al abrir EchidnaML el programa no detecta la placa se puede deber a:

1. EchidnaBlack no tiene instalado el Firmware Standardfirmata:

Aunque el firmware viene instalado de serie alguien puede haber escrito otro programa. Si es así debemos instalar el programa Standardfirmata tal como se especifica en el punto 6 de esta guía.

2. Nuestro ordenador no reconoce el puerto USB al que se conecta EchidnaML:

Para que EchidnaBlack se pueda comunicar con nuestro PC es necesario que nuestro sistema operativo (SO) le dé permiso de acceso al puerto serie (USB) y que tenga el driver del controlador de comunicación (CH430) instalado.

2.1 Driver de comunicación:

Echidna Black utiliza el chip CH430E, por lo que dependiendo del SO necesitarás instalar el controlador "Driver CH341"

- GNU Linux: En caso de que seas usuario Linux, no debería ser necesario instalar el driver. Si necesitas el driver para GNU Linux.
- MAC: Aquí tienes acceso al driver para MAC.
- Windows: Aquí tienes acceso al driver para Windows.

2.2 Permiso de acceso al puerto serie:

Dependiendo del SO es necesario dar o no permisos de acceso al puerto serie. Si ya has instalado el IDE de Arduino, estos permisos deberían estar ya dados.

- **GNU Linux:** Si no no tuvieras acceso al puerto serie puede que tengas que darle permiso desde una terminal usando el comando: sudo usermod -a -G dialout «usuario». Luego es necesario salir de la sesión y volver a entrar para que los cambios se hagan efectivos.
- **MACOS**: por defecto ya tenemos acceso al puerto serie.
- **Windows:** por defecto ya tenemos acceso al puerto serie.

8.4 Qué tensiones son seguras para la placa

Alimentación por USB-C (Recomendada para la mayoría de usos)

Esta es la forma más sencilla, y común de usar la placa, mediante un cable USB-C conectado a un ordenador o a un cargador.

Características:

- La placa recibe una tensión de 5 V y suficiente energía para las funciones básicas.
- Tensión de uso: Toda la placa recibe una tensión regulada y estable de 5V.
- Límite de energía: Es suficiente para las funciones básicas y la mayoría de sensores pequeños (generalmente con un límite de 500 mA).
- Protección: Cuenta con un fusible rearmable que se desconecta si hay un consumo excesivo.

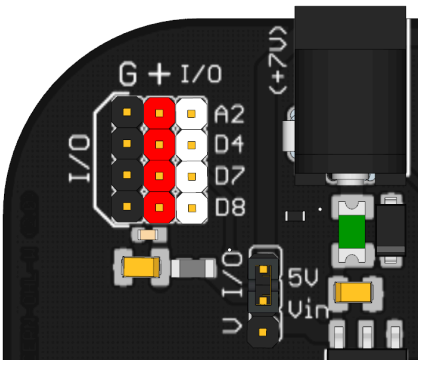
Alimentación por Jack (Recomendada para proyectos grandes o autónomos)

La usamos cuando queremos conectar elementos con mucha potencia, como varios servomotores, o para proyectos autónomos.

Características:

- Tensión de uso: La placa recibe una tensión regulada y estable de 5V.
- Límite de energía: Es suficiente para las funciones básicas y la mayoría de sensores (generalmente con un límite de 1000 mA).
- Protección: Cuenta con un fusible rearmable que se desconecta si hay un consumo excesivo.

Selector de Alimentación I/O



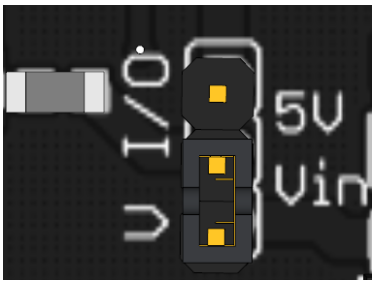
Este **jumper** te permite elegir qué tensión quieres enviar a los pines de entrada/salida de la placa (I/O, concretamente A2, D4, D7, D8) que usarán los componentes externos.

- **Selector alimentación 5V:**

En el caso de querer alimentar las I/O desde los 5 Volts procedentes del regulador integrado, el selector tiene que estar colocado como indica la imagen de la izquierda. Aconsejado para sensores externos que necesiten una tensión estabilizada.

⚠️ ¡No utilizar la alimentación 5 Volts cuando los dispositivos conectados consuman más de 500 mA!. De lo contrario sobrepasaríamos la capacidad del regulador.

- **Selector alimentación Vin (Alimentación externa):**



  Aconsejado para alimentar servos u otros dispositivos conectados a I/O que necesiten una tensión mayor de 5 Volts.

- ⚠️ **¡ATENCIÓN!**
  Debes ser **extremadamente cuidados@**. Si tu alimentador externo tiene una tensión superior a la que soportan tus componentes conectados a I/O (que a menudo son 5 Volts), **¡puedes quemarlos inmediatamente!**

8.5 Puedo conectar la placa una vez abierto programa

Lo más recomendable es conectar la placa EchidnaBlack2 al ordenador antes de abrir el entorno de programación EchidnaML. Esto asegura una detección rápida y que no perdamos el trabajo realizado.

Si ya has abierto el programa EchidnaML y quieres conectas la placa posteriormente:

1. Conecte la placa al ordenador mediante el cable USB.
2. Para iniciar la detección, debe hacer clic en el icono USB (o el icono de conexión) dentro de la interfaz.

⚠️ **Advertencia: Guardar el Trabajo**

Debe tener en cuenta que el proceso de detección y conexión reinicia el entorno de trabajo, provocando la pérdida de cualquier proyecto no guardado.

Por esta razón, siempre guarde su proyecto en el ordenador antes de iniciar el proceso de detección para evitar la pérdida de trabajo y poder recuperarlo.

9. CARACTERÍSTICAS TÉCNICAS DE LA PLACA

Microcontrolador: AtMega 328P, 16MHz. Compatible con Arduino Nano.

Sensores incorporados:

- Joystick X,Y lineal
- Sensor de luz LDR sensible 400 - 600LUX a 540nm
- Sensor de temperatura -40°C to +125°C 10mV/ºC
- Acelerómetro X-Y-Z ±2g (I2C “0x18h”)
- Micrófono omnidireccional 100 - 20KHZ.
- Pulsadores 12mm 1.27 - 2.55N
- Entradas MKMK Conductivas

Actuadores incorporados:

- LEDes 5mm Rojo, naranja y Verde.
- LED RGB 65535 colores
  - Rojo (619 - 624nm)
  - Verde (520 - 540nm)
  - Azul (460 - 480nm
- Zumbador 2300Hz, 85dB a 10cm
  - Conexión Jack (audio) 3.5mm
  - Control de volumen con potenciómetro lineal.

Conectores incorporados:

- USB-C (Programación/comunicación)
- Conector 6 pines para módulo Bluetooth HC-05…
  - K, 5V, GND, RX, TX, NC
- Conector 4 pines I^2C
  - GND, 5V, SCL, SDA
- Conector I/O
  - GND, + V, A2
  - GND, + V, D4
  - GND, + V, D7
  - GND, + V, D8
- Conexión Jack de alimentación 2.5mm, 6,4mm
  - (+V), GND
- Conector 6 pines ICSP

Consumo: Conexión por USB 0,150W, Alimentación externa 0.160W.

10. REFERENCIAS

En esta sección, encontrará enlaces y recursos adicionales para quienes deseen ampliar la información sobre los temas tratados en esta Guía

Echidna: https://echidna.es/

- EchidnaBlack^2: características del Hardware.
- Descargar el programa EchidnaML
- Empieza a trabajar con EchidnaML y EchidnaScratch: vídeos explicativos
- Manual en versión PDF actualizada
- Imprimir la carcasa:
- REA: Recursos educativos abiertos realizados con exelearning:
- Diapositivas para uso en clase Primaria y Secundaria

LearningML: https://web.learningml.org/

- Manual de learningML
- LearnigML online

Scratch: https://scratch.mit.edu/

- Tutorial de introducción
- Explorar proyectos de iniciación

Arduino:

- Descargar IDE Arduino
- Instalar IDE de Arduino:

11. LICENCIA