Robótica Sphero Mini

Aprende a programar este simpático robot: Introducción 1 Primero: APP de Jugar 2 Segundo: APP de Programar 2.1 Registrarse 2.2 Crear clase 2.3 Encender 2.4 Apagar 3 A PROGRAMAR!! 3.1 Giro-mensaje 3.2...

- 1 Introducción
  - Portada
  - Pensamiento computacional
  - Robótica y accesibilidad
  - 1.1 Primero: APP de Jugar
  - 1.2 Segundo: APP de Programar
  - 1.3 Encender
  - 1.4 Apagar
- 2 A programar!!
  - 2.1 Giro-mensaje
  - 2.2 Malabares
  - 2.3 Cuadrado
  - 2.4 ¿Ganaré la lotería?
  - 2.5 Actividades de otros
  - 2.6 Conclusiones
- 3 Para saber más ...
  - 3.2 Muro

Copyright 2026

- Créditos

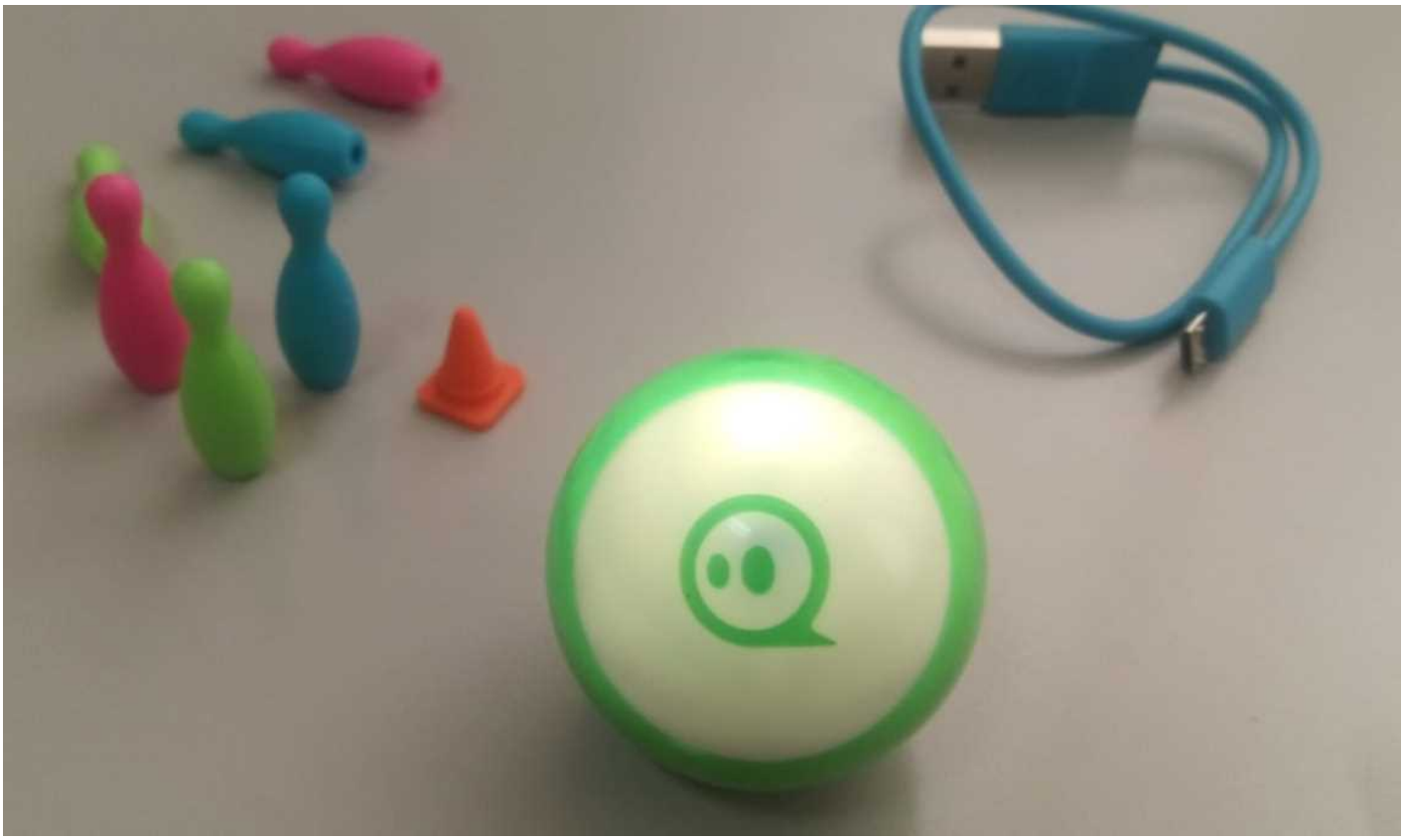
Copyright 2026

1 Introducción

Copyright 2026

1 Introducción

Portada



Copyright 2026

1 Introducción

Pensamiento computacional

¿Dónde se encaja este robot?¿se puede comparar este robot con otros robots de otros cursos que hacemos desde CATEDU?

Esta es la hoja de ruta que proponemos, no se tiene que tomar al pie de la letra, pero intenta ayudar al profesorado que tenga una visión global de tanta oferta:

Como se puede ver **SPHERO MINI** tiene la ventaja de tener un precio razonable, resistente y ocupar poco espacio dentro del rango de programación en primaria.

Guía orientativa

https://docs.google.com/presentation/d/e/2PACX-1vQHiZvv1cGHet7eXVy-QcECY4Lj0k0l7ntDi8MevRWHQX-9myA0bfR5IofMeuGZkWD0Hw-Ob-MGoco_/embed?start=trueloop=true&delayms=3000

> Tenemos un **grupo Telegram Robótica Educativa en Aragón**, https://t.me/roboticaeducativaaragon



Copyright 2026

1 Introducción

Robótica y accesibilidad

1.- Introducción

Durante mucho tiempo la robótica fue patrimonio de personas y/o instituciones con alta capacidad económica (podían adquirir las placas con microcontroladores comerciales) y capacidad intelectual (podían entender y programar el funcionamiento de las mismas) siempre dentro de los límites establecidos por las marcas comerciales y lo que pudieran “desvelar” de su funcionamiento, vigilando siempre que la competencia no “robara” sus secretos y “copiara” sus soluciones.

Todo esto saltó por los aires en torno a 2005 con la irrupción de un grupo de profesores y estudiantes jóvenes, que decidieron romper con esta dinámica, tratando de poner a disposición de su alumnado microcontroladores económicamente accesibles y que les permitieran conocer su funcionamiento, sus componentes, e incluso replicarlos y mejorarlos. Nacía **Arduino** y el concepto de **Hardware Open Source**. Detrás de este concepto se encuentra la **accesibilidad universal.** En un proyecto Open Source todo el mundo puede venir, ayudar y contribuir, minimizando barreras económicas e intelectuales.

Arduino traslada al hardware un concepto ya muy conocido en el ámbito del software, como es el **software open source o software libre.**



Software libre

Cuando los desarrolladores de software terminan su creación, tienen múltiples posibilidades de ponerlo a disposición de las personas, y lo hacen con condiciones específicas especificadas en una licencia. Esta licencia es un contrato entre el creador o propietario de un software y la persona que finalmente acabará utilizando este software. Como usuarios, es nuestro deber

conocer las condiciones y permisos con las que el autor ha licenciado su producto, para conocer bajo qué condiciones podemos instalar y utilizar cada programa.

Existen muchas posibilidades de licencias: software privativo, comercial, freeware, shareware, etc.. Nos centraremos aquí en la de software libre.

GNU (https://www.gnu.org) es una organización sin ánimo de lucro que puso una primera definición disponible de lo que es software libre: Software libre significa que los usuarios del software tienen libertad (la cuestión no es el precio). Desarrollaron el sistema operativo GNU para que los usuarios pudiesen tener libertad en sus tareas informáticas. Para GNU, el software libre implica que los usuarios tienen las cuatro libertades esenciales:

1. ejecutar el programa.
2. estudiar y modificar el código fuente del programa.
3. redistribuir copias exactas.
4. distribuir versiones modificadas.

En otras palabras, el software libre es un tipo de software que se distribuye bajo una licencia que **permite a los usuarios utilizarlo, modificarlo y distribuirlo libremente**. Esto significa que los usuarios tienen libertad de ejecutar el software para cualquier propósito, de estudiar cómo funciona el software y de adaptarlo a sus necesidades, de distribuir copias del software a otros usuarios y de mejorar el software y liberar las mejoras al público.

El software libre se basa en el principio de la libertad de uso, y no en el principio de la propiedad. Esto significa que los usuarios tienen la libertad de utilizar el software de la manera que deseen, siempre y cuando no violen las condiciones de la licencia. El software libre es diferente del software propietario, que es el software que se distribuye con restricciones en su uso y modificación. El software propietario suele estar protegido por derechos de autor y solo se puede utilizar bajo los términos y condiciones especificados por el propietario del software.

Recomendamos la visualización de este video para entender mejor el concepto.

https://www.youtube.com/embed/nlDVZ816zoI

Más adelante, entorno a 2015, en Reino Unido, surgiría también la placa **BBC Micro:bit**, con la misma filosofía de popularizar y hacer accesible en este caso al alumnado de ese país la programación y la robótica. También hablaremos de ella.

Copyright 2026

2.- ARDUINO o LA ROBÓTICA ACCESIBLE

Arduino es una **plataforma de hardware y software libre**.

Hardware libre

Esto significa que tanto la placa Arduino como el entorno de desarrollo integrado (IDE) son de código abierto. Arduino permite a los usuarios utilizar, modificar y distribuir tanto el software como el hardware de manera libre y gratuita, siempre y cuando se respeten las condiciones de las licencias correspondientes.

El hardware libre es un tipo de hardware cuya **documentación y diseño están disponibles de manera gratuita y libre** para su modificación y distribución. Esto permite a los usuarios entender cómo funciona el hardware y adaptarlo a sus necesidades, así como también crear sus propias versiones modificadas del hardware.

Arduino surge como solución al **elevado precio de los microcontroladores** allá por el año 2005. En el ámbito de la educación, los microcontroladores solo se utilizaban en la etapa universitaria, y su coste era tan elevado que muchos proyectos de fin de carrera se quedaban únicamente en prototipos virtuales ya que las universidades no podían proveer a cada estudiante con un microprocesador, contando además que en el propio proceso de experimentación lo más habitual era que una mala conexión hiciera que se rompieran. Otro **gran inconveniente era la dificultad de la programación**. Cada fabricante entregaba su manual de programación, lo que hacía que de unos a otros no hubiera un lenguaje estándar, y la consecuente dificultad de interpretación. Además, su programación era a bajo nivel en lenguaje máquina. Generar una simple PWM requería una ardua y minuciosa secuenciación que podía llevar varias horas hasta conseguir el resultado deseado. Por este motivo, el enfoque de Arduino desde el principio fue ser Open Source tanto en hardware como en software. El desarrollo del hardware fue la parte más sencilla. Orientado a educación, sufre algunas modificaciones frente a los microprocesadores existentes para hacer más fácil su manejo y accesibilidad a cualquier sensor o actuador. El mayor esfuerzo se entregó en todas las líneas de código que hacían posible que ya no hubiera que programar a bajo nivel gracias al IDE de Arduino que incluía bibliotecas y librerías que estandarizaban los procesos y hacían tremendamente sencillo su manejo. Ahora el alumnado para mover un motor, ya no tenía que modificar las tramas de bits del procesador una a una, sino que bastaba con decir que quería moverlo en tal dirección, a tal velocidad, o a equis grados.

Acabábamos de pasar de unos costes muy elevados y una programación muy compleja a tener una **placa accesible, open source y de bajo coste** que además hacía muy **accesible su programación y entendimiento**, características fundamentales para su implantación en

Copyright 2026

educación, hasta tal punto que su uso ya no era exclusivo de universidades, sino que se extiende a la educación secundaria.



Este hecho es fundamental para el desarrollo del Pensamiento Computacional en el aula observándose que su accesibilidad y beneficios son tales, que alcanzan a **centros con alumnado de toda tipología** como la aplicación del pensamiento computacional y robótica en aulas con alumnos de necesidades especiales. Una vez más, aparece el concepto de accesibilidad asociado a esta filosofía Open Source.

A este respecto, recomendamos la lectura de este interesante blog, que tiene por título: ROBOTIQUEAMOS…" Experiencia de aproximación a la robótica en Educación Especial (CPEE ÁNGEL RIVIÈRE). También recomendamos los trabajos robótica en Educación Especial (CPEE ÁNGEL RIVIÈRE): http://zaragozacpeeangelriviere.blogspot.com/search/label/ROB%C3%93TICA



Igualmente, la aparición de Arduino supone una gran facilidad para la aplicación de la robótica y la programación en la atención temprana, donde son numerosas sus aplicaciones desde ayudar a mitigar el déficit de atención en jóvenes autistas, hasta ayudar a socializar a los alumnos con dificultades para ello, o ayudar a alumnos de altas capacidades a desarrollar sus ideas.

Por otro lado su accesibilidad económica lo ha llevado a popularizarse en países de **todo el mundo**, especialmente en aquellos cuyos sistemas educativos no disponen en muchas ocasiones de recursos suficientes, lo que supone en la práctica una **democratización del conocimiento y**

superación de brecha digital.

Filosofía del Arduino ver vídeo

Arduino y su IDE son la primera solución que aparece en educación con todas las ventajas que hemos enumerado, y esto hace que todos los nuevos prototipados y semejantes tengan algo en común, siempre son compatibles con Arduino

Para entender bien la filosofía de Arduino y el hardware libre, os recomendamos este documental de 30 minutos. Arduino the Documentary

https://player.vimeo.com/video/18390711?h=b5844e7753

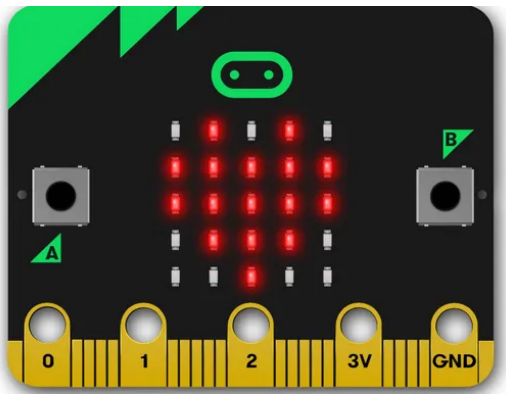
Scratch: software libre para el desarrollo del pensamiento computacional

Scratch es un lenguaje de programación visual desarrollado por el grupo Lifelong Kindergarten del MIT Media Lab. Scratch es un software libre. Esto significa que está disponible gratuitamente para todos y que se distribuye bajo una licencia de software libre, la Licencia Pública General de Massachusetts (MIT License). Esta licencia permite a los usuarios utilizar, modificar y distribuir el software de manera libre, siempre y cuando se respeten ciertas condiciones. Entre otras cosas, la licencia de Scratch permite a los usuarios utilizar el software para cualquier propósito, incluyendo fines comerciales. También permite modificar el software y distribuir las modificaciones, siempre y cuando se incluya una copia de la licencia y se indique que el software ha sido modificado. En resumen, Scratch es un software libre que permite a los usuarios utilizar, modificar y distribuir el software de manera libre y gratuita, siempre y cuando se respeten las condiciones de la licencia. De hecho, gracias a que está licenciado de esta forma, han surgido decenas de variaciones de Scratch para todo tipos de propósitos, eso sí, siempre educativos y relacionados con las enseñanzas de programación y robótica

3. BBC micro:bit y la Teoría del Cambio

Copyright 2026

BBC micro:bit, a veces escrito como Microbit o Micro Bit, es un pequeño ordenador del tamaño de media tarjeta de crédito, creado en 2015 por la BBC con el fin de promover el desarrollo de la robótica y el pensamiento computacional entre la población escolar del Reino Unido. Actualmente s es de escolares de 7 a 16 años de más de 60 países.



Tarjeta BBC micro:bit V1. Fuente: https://microbit.org. CC BY-SA 4.0.

Aunque el proyecto fue iniciado por la BBC, su desarrollo fue llevado a cabo por 29 socios tecnológicos de primera línea. Por ejemplo, la implementación del Bluetooth integrado en la tarjeta corrió a cargo de la fundación propietaria de la marca, Bluetooth SIG, una asociación privada sin ánimo de lucro.

El hardware y el software resultantes son 100% abiertos, y están gestionados por una fundación sin ánimo de lucro que comenzó a funcionar en el año 2016, la Micro:bit Educational Foundation. La fundación basa sus actuaciones en su Teoría del Cambio,

Teoría del cambio y más sobre microbit

Teoría del cambio puede resumirse en tres principios:

- El convencimiento de que la capacidad de comprender, participar y trabajar en el mundo digital es de vital importancia para las oportunidades de vida de una persona joven.
- La necesidad de emocionar y atraer a las personas jóvenes por medio de BBC micro:bit, especialmente a las que podrían pensar que la tecnología no es para ellas.
- Diversificar a los estudiantes que eligen las materias STEM a medida que avanzan en la escuela y en sus carreras, para hacer crecer una fuente diversa de talento, impulsando la equidad social y contribuyendo a crear una tecnología mejor.

Para desarrollar sus principios, la fundación trabaja en tres líneas de acción:

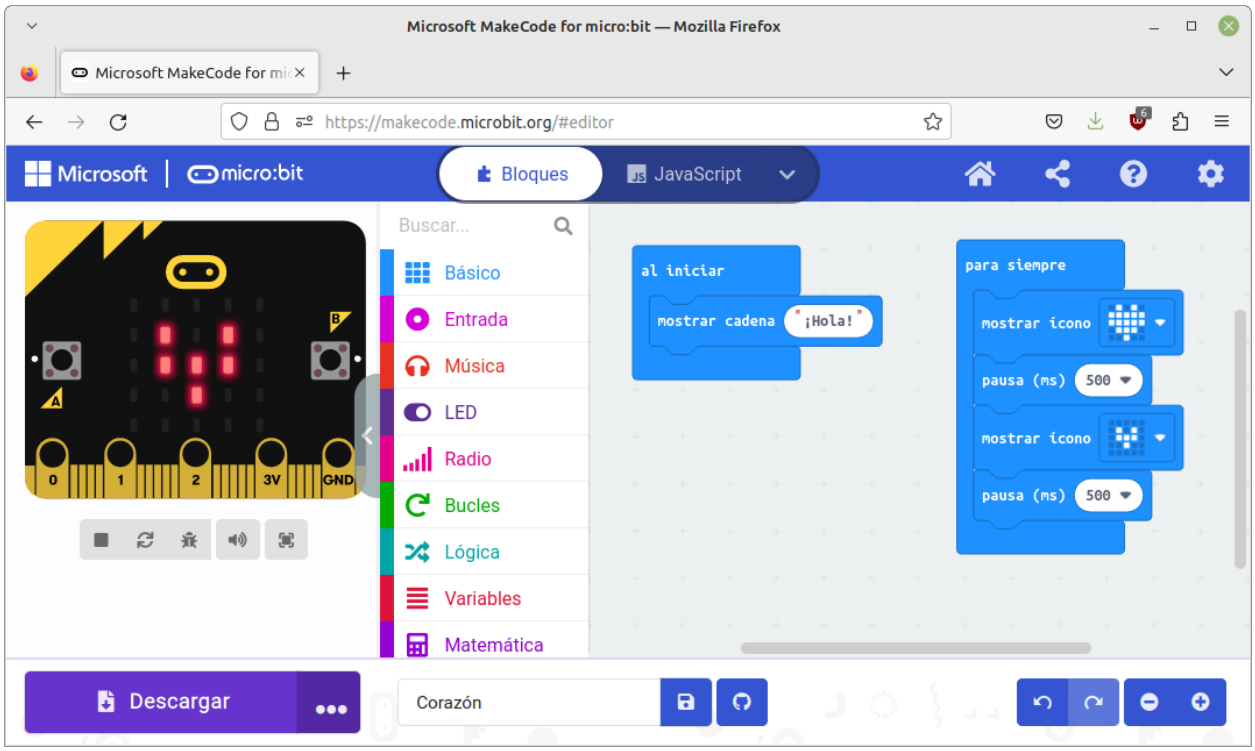
- El desarrollo de hardware y software que contribuyan a despertar el entusiasmo en las personas jóvenes hacia la tecnología y hacia las oportunidades que presenta.
- La creación de recursos educativos gratuitos y fáciles de usar que permitan al profesorado enseñar de forma atractiva y creativa.
- La colaboración con entidades asociadas que compartan una misma visión para ofrecer programas educativos de alto impacto en todo el mundo.

> Uno de los objetivos de la Micro:bit Educational Foundation es llegar a 100 millones de escolares en todo el mundo.

En correspondencia con las líneas de acción y con los principios expuestos, el sistema resultante es muy económico: tanto las placas como los accesorios producidos por terceras empresas tienen un precio muy contenido. Además, dado el carácter abierto del proyecto, están disponibles algunos clones totalmente compatibles, como Elecrow Mbits o bpi:bit. Estos clones son incluso más potentes y económicos que la placa original.

El universo micro:bit destaca por su **alta integración de software y hardware**: basta un clic de ratón para cargar las librerías necesarias para que funcione cualquier complemento robótico, como sensores, pantallas, tarjetas de Internet de las Cosas, robots, casas domóticas, etc.

La programación de la placa se realiza desde un ordenador a través de un navegador cualquiera, estando disponibles **12 lenguajes de programación**. De nuevo, por ser un sistema abierto, existen múltiples soluciones de programación, aunque las más común es MakeCode.



Captura de pantalla del editor MakeCode, https://makecode.microbit.org/#.

El sitio web MakeCode permite programar con bloques y también en Python y en Java, traduciendo de un lenguaje a otro instantáneamente. No se necesita ningún registro en la plataforma para poder programar.

Los programas también pueden guardarse descargados en el ordenador compilados en código de máquina. Al subir de nuevo el programa al editor, se realiza una decompilación automática al lenguaje de bloques, Python o Java. Los programas guardados en código de máquina se pueden cargar directamente en micro:bit, que en el escritorio de un ordenador se maneja como una simple unidad de memoria USB.

MakeCode contiene además múltiples recursos como tutoriales, vídeos, fichas de programación, cursos para el profesorado, ejemplos y propuestas de proyectos y experimentos, todo ello en varios idiomas y clasificado por edades desde los 7 años.

Otra solución muy usada para programar micro:bit es MicroPython, creada por Python Software Foundation, otra organización sin ánimo de lucro.

MicroCode permite que los más pequeños, a partir de los 6 años de edad, programen micro:bit mediante un sistema de fichas dispuestas en líneas de acción. Están disponibles un tutorial introductorio en 20 idiomas, una guía del usuario y muchos ejemplos. El proyecto es de código abierto.

Copyright 2026

Micro:bit también es programable en **Scratch** con sólo añadir una extensión al editor.

Todos los entornos de desarrollo descritos disponen de un simulador de micro:bit, por lo que ni siquiera resulta necesario disponer de una tarjeta física para aprender a programar.

Una vez realizada la programación, la placa y sus complementos pueden funcionar desconectados del ordenador por medio de un cargador de móvil, una batería externa o un simple par de pilas alcalinas.

Versiones y características de micro:bit

A pesar de su pequeño tamaño, micro:bit es un sistema potente. Existen dos versiones de la placa. La más moderna, llamada micro:bit V2, tiene las siguientes características:

- Procesador de 64 MHz.
- 512 KB de RAM Flash y 128 KB de RAM.
- Matriz de 5 x 5 LED rojos.
- Dos pulsadores mecánicos y un tercer pulsador de apagado y reset.
- Un pulsador táctil.
- Micrófono y altavoz.
- Acelerómetro y brújula.
- Sensores de luz y de temperatura.
- Comunicación con otras placas por Bluetooth de bajo consumo.
- Alimentación a 3 V o por USB.
- 25 pines de entradas y salidas para conectar motorcitos, sensores, placas de Intenet de las Cosas, robots y, en general, cualquier otro tipo de accesorio.
- 200 mA de intensidad de corriente disponibles en las salidas para alimentar accesorios.

4.- LA IMPORTANCIA DEL OPEN SOURCE / CÓDIGO ABIERTO EN EDUCACIÓN

La creación, distribución, modificación y redistribución del hardware y software libre así como su utilización, están asociados a una serie de valores que deberían ser explicados en la escuela a nuestros alumnos para dar una alternativa a la versión mercantilista de que cualquier creación es creada para obtener beneficios económicos.

Copyright 2026

En GNU, pusieron especial énfasis en la difusión del software libre en colegios y universidades, promoviendo una serie de valores fundacionales:

Valores GNU

Compartir

El código fuente y los métodos del hardware y software libre son parte del conocimiento humano. Al contrario, el hardware software privativo es conocimiento secreto y restringido. El código abierto no es simplemente un asunto técnico, es un asunto ético, social y político. Es una cuestión de derechos humanos que la personas usuarias deben tener. La libertad y la cooperación son valores esenciales del código abierto. El sistema GNU pone en práctica estos valores y el principio del compartir, pues compartir es bueno y útil para el progreso de la humanidad. Las escuelas deben enseñar el valor de compartir dando ejemplo. El hardware y software libre favorece la educación pues permite compartir conocimientos y herramientas.

Responsabilidad social

La informática, electrónica, robótica... han pasado a ser una parte esencial de la vida diaria. La tecnología digital está transformando la sociedad muy rápidamente y las escuelas ejercen una influencia decisiva en el futuro de la sociedad. Su misión es preparar al alumnado para que participen en una sociedad digital libre, mediante la enseñanza de habilidades que les permitan tomar el control de sus propias vidas con facilidad. El hardware y el software no debería estar bajo el poder de un desarrollador que toma decisiones unilaterales que nadie más puede cambiar.

Independencia

Las escuelas tienen la responsabilidad ética de enseñar la fortaleza, no la dependencia de un único producto o de una poderosa empresa en particular. Además, al elegir hardware y software libre, la misma escuela gana independencia de cualquier interés comercial y evita permanecer cautiva de un único proveedor. Las licencias de hardware y software libre no expiran

Aprendizaje

Con el open source los estudiantes tienen la libertad de examinar cómo funcionan los dispositivos y programas y aprender cómo adaptarlos si fuera necesario. Con el software libre se aprende también la ética del desarrollo de software y la práctica profesional.

Copyright 2026

Ahorro

Esta es una ventaja obvia que percibirán inmediatamente muchos administradores de instituciones educativas, pero se trata de un beneficio marginal. El punto principal de este aspecto es que, por estar autorizadas a distribuir copias de los programas a bajo costo o gratuitamente, las escuelas pueden realmente ayudar a las familias que se encuentran en dificultad económica, con lo cual promueven la equidad y la igualdad de oportunidades de aprendizaje entre los estudiantes, y contribuyen de forma decisiva a ser una escuela inclusiva.

Calidad

Estable, seguro y fácilmente instalable, el software libre ofrece una amplia gama de soluciones para la educación.

Para saber más

En los años 90, era realmente complicado utilizar un sistema operativo Linux y la mayoría de la cuota del mercado de los ordenadores personales estaba dominada por Windows. Encontrar drivers de Linux para el hardware que tenía tu equipo era casi una quimera dado que las principales compañías de hardware y de software no se molestaban en crear software para este sistema operativo, puesto que alimentaba la independencia de los usuarios con respecto a ellas mismas.

Afortunadamente, y gracias a la creciente presión de su comunidad de usuarios, estas situaciones pertenecen al pasado, y las compañías fabricantes de hardware han tenido que variar el rumbo. Hoy en día tenemos una gran cantidad de argumentos en los que nos podemos basar para dar el salto hacia cualquier sistema operativo basado en Linux. Tal y como podemos leer en educacionit.com, podemos encontrar las siguientes ventajas:

- Es seguro y respeta la privacidad de los usuarios: Aunque hay compañías linuxeras, como Oracle, Novell, Canonical, Red Hat o SUSE, el grueso de distribuciones y software Linux está mantenido por usuarios y colectivos sin ánimo de lucro. De esta forma, podemos confiar en que una comunidad que tiene detrás millones de usuarios, pueda validar el código fuente de cualquier de estas distribuciones, asegurándonos la calidad de las mismas, compartir posibles problemas de seguridad, y sobre todo, estar bien tranquilos con la privacidad y seguridad de nuestros datos e información

personal, aspecto que debería ser crítico y determinante a la hora de trabajar con los datos de menores de edad en las escuelas y colegios.

- Es ético y socialmente responsable: La naturaleza de Linux y su filosofía de código abierto y libre hace posible que cualquier usuario con conocimientos pueda crear su propia distribución basada en otras o probar las decenas de versiones que nos podemos encontrar de una distribución Linux. Este es el caso de Ubuntu por ejemplo. Gracias a esta democratización de los sistemas operativos, incluso han podido aparecer en nuestras vidas nuevos dispositivos basados en software y hardware libre como Arduino y Raspberry Pi.
- Es personalizable: el código abierto permite su estudio, modificación y adaptación a las necesidades de los diferentes usuarios, teniendo así no un único producto sino una multiplicidad de distribuciones que satisfacen las necesidades de los diferentes colectivos a los que se dirijan. Especialmente útiles son las distribuciones educativas libres, que pueden ser adaptadas a las necesidades de las escuelas.
- Está basado en las necesidades de los usuarios y no en las de los creadores de hardware y software
- Es gratis. La mayoría de las distribuciones Linux son gratuitas y de libre descarga
- Es fácil de usar. Una de las barreras que durante años ha evitado a muchos usar Linux es su complejidad. Las distribuciones orientadas al consumo doméstico cumplen los estándares de simplicidad y necesidades que cualquier usuario sin conocimientos de tecnología pueda necesitar. El entorno gráfico es sencillo, intuitivo, e incluso se puede customizar para que se pueda parecer a los más conocidos como Windows y MacOS. Además, vienen con la mayoría de aplicaciones que cualquier usuario puede necesitar: ofimáticas, edición de audio y vídeo y navegación por Internet.
- Es suficiente. Tiene su propio market de aplicaciones. Como el resto de sistemas operativos ya sea para ordenadores o dispositivos móviles, también podemos encontrar un lugar único donde poder descargar cientos de aplicaciones para todos los gustos y necesidades.

Por estas razones, el software libre se ha expandido por toda la comunidad educativa en los últimos años de manera exponencial. Un buen ejemplo de lo que estamos hablando es **Bookstack**, este sistema de edición de contenidos para cursos que utiliza Aularagón así como el uso de **Moodle** como plataforma de enseñanza y aprendizaje. En cuanto a sistema operativo para ordenadores, en Aragón disponemos de nuestra propia distribución Linux: Vitalinux EDU. Tal y como podemos leer desde su página web: **Vitalinux EDU (DGA)** es la distribución Linux elegida por el Gobierno de Aragón para los centros educativos. Está basada en Vitalinux, que se define como un proyecto para llevar el Software Libre a personas y organizaciones facilitando al máximo su instalación, uso y mantenimiento. En concreto Vitalinux EDU (DGA) es una distribución Ubuntu (Lubuntu) personalizada para Educación, "tuneada" por los requisitos y necesidades de los propios usuarios de los centros y adaptada de forma personalizada a cada

centro y a la que se ha añadido una aplicación cliente Migasfree. De ésta forma, obtenemos:

1. Un **Sistema Ligero.** Permite "revivir" equipos obsoletos y "volar" en equipos modernos. Esto garantiza la sostenibilidad de un sistema que no consume recursos de hardware innecesariamente ni obliga a la sustitución del hardware cada poco tiempo en esa espiral de obsolescencia programada en la que se ha convertido el mercado tecnológico.
2. **Facilidad en la instalación y el uso** del sistema mediante programas personalizados.
3. Un Sistema que **se adapta al centro** y/o a cada aula o espacio, y no un centro que se adapta a un Sistema Operativo.
4. **Gestión de equipo y del software de manera remota** y desatendida mediante un servidor Migasfree.
5. **Inventario** de todo el hardware y software del equipo de una forma muy cómoda.
6. Soporte y apoyo de una **comunidad** que crea, comparte e innova constantemente.

Copyright 2026

1 Introducción

1.1 Primero: APP de Jugar

Lo primero que podemos hacer es abrir el paquete, ver el contenido y ... jugar un poco con la APP **Sphero Play**



https://www.youtube.com/embed/D4XBH5ggq-w

O en español:

https://www.youtube.com/embed/kBrcEWNas5Q

1 Introducción

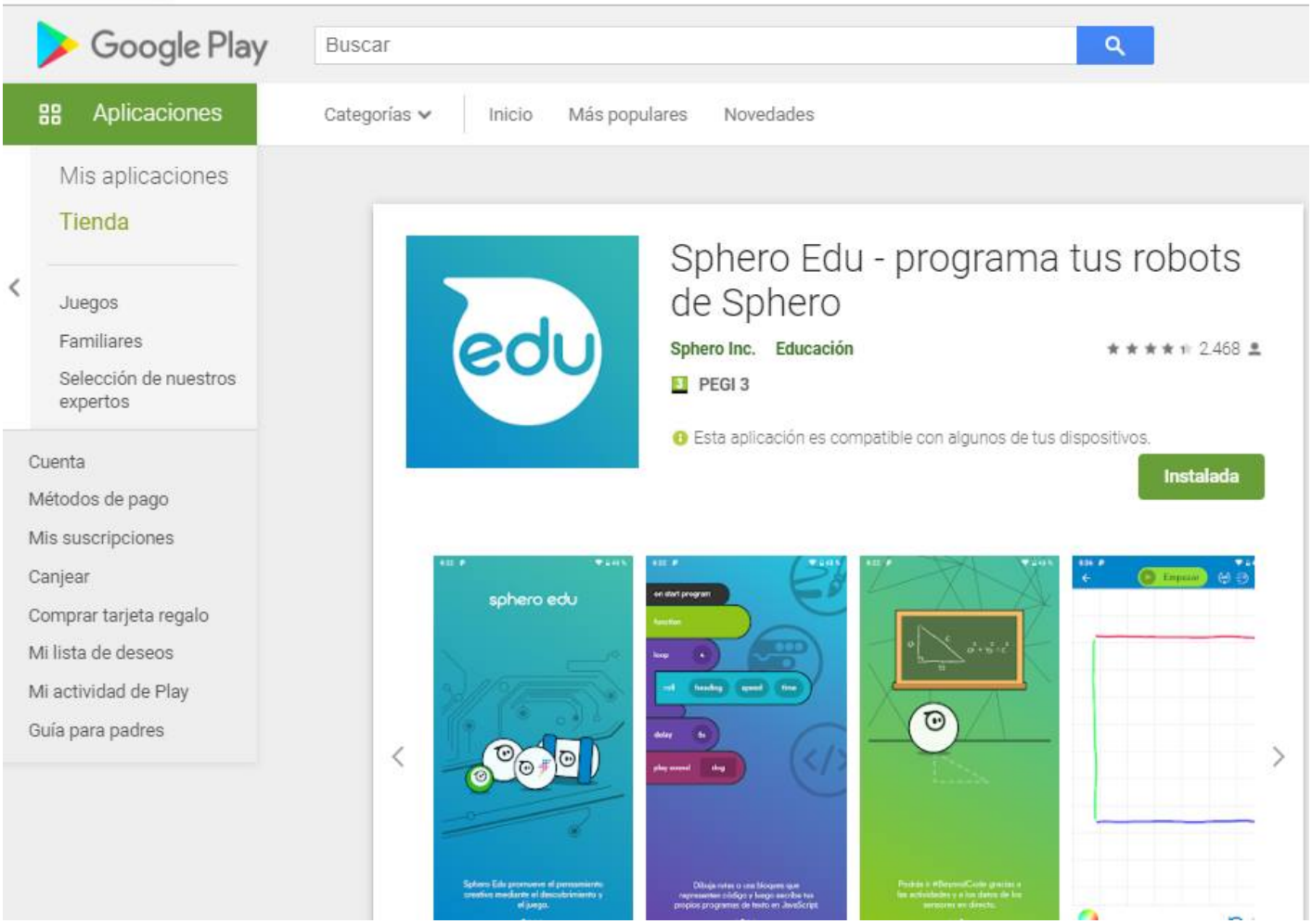
1.2 Segundo: APP de Programar

Segundo Programar

Pero lo que nos interesa no es jugar, sino aprender el prensamiento computacional

Las aplicación para el móvil es distinta: SpheroEdu

Copyright 2026



Con el ordenador Windows

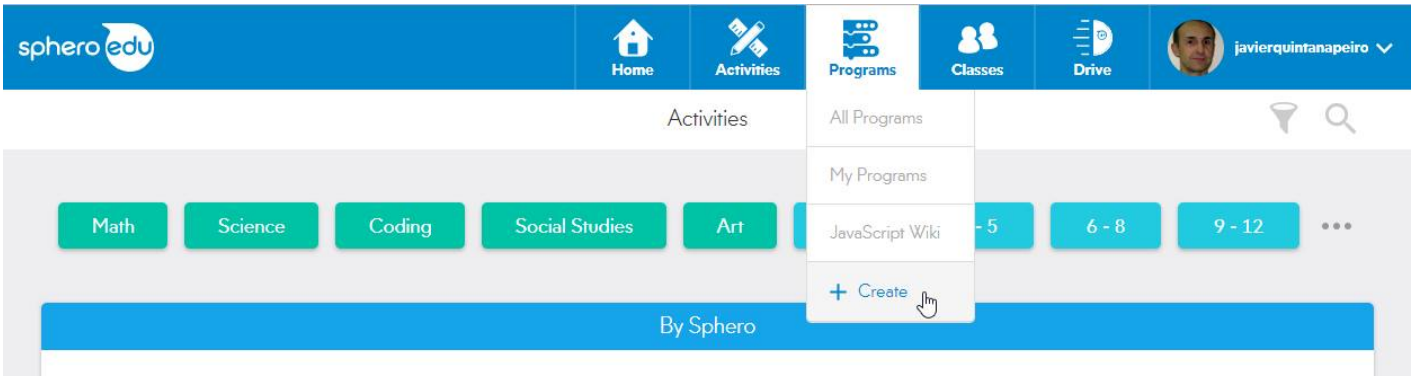
En Windows nos podemos descargar su app que se instala y funciona perfectamente si el equipo tiene Bluetooth



> Para otros sistemas operativos, buscar en Internet

On line no funciona

Si entras en https://edu.sphero.com/



Puedes crear tus programas, compartirlos, gestionar tus clases:



PERO NO PUEDES CONECTARTE CON EL ROBOT por lo tanto no podemos programar con el robot ¿entonces para qué sirve esta web?

Robot Compatibilty

Sphero Robot	Year Launched	Mobile Apps			Desktop Apps			Website
		iOS	Android	Fire OS	Windows	macOS	Chrome OS	
Sphero RVR	2019	●	●	●	●	●	●	
Sphero BOLT	2018	●	●	●	●	●	●	
Sphero SPRK+	2016	●	●	●	●	●	●	
Sphero Mini	2017	●	●	●	●	●	●	

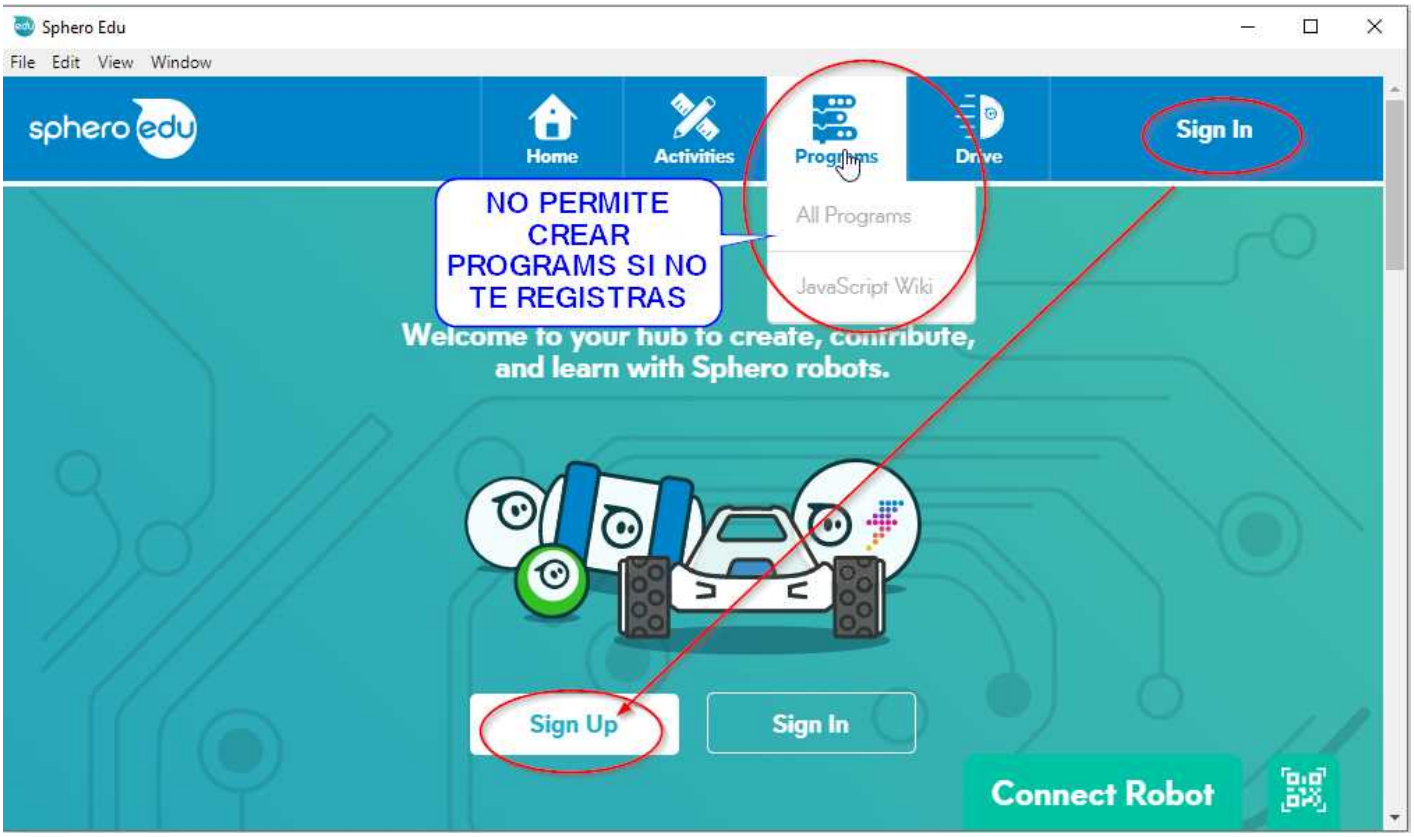
ver artículo

Nuestro consejo

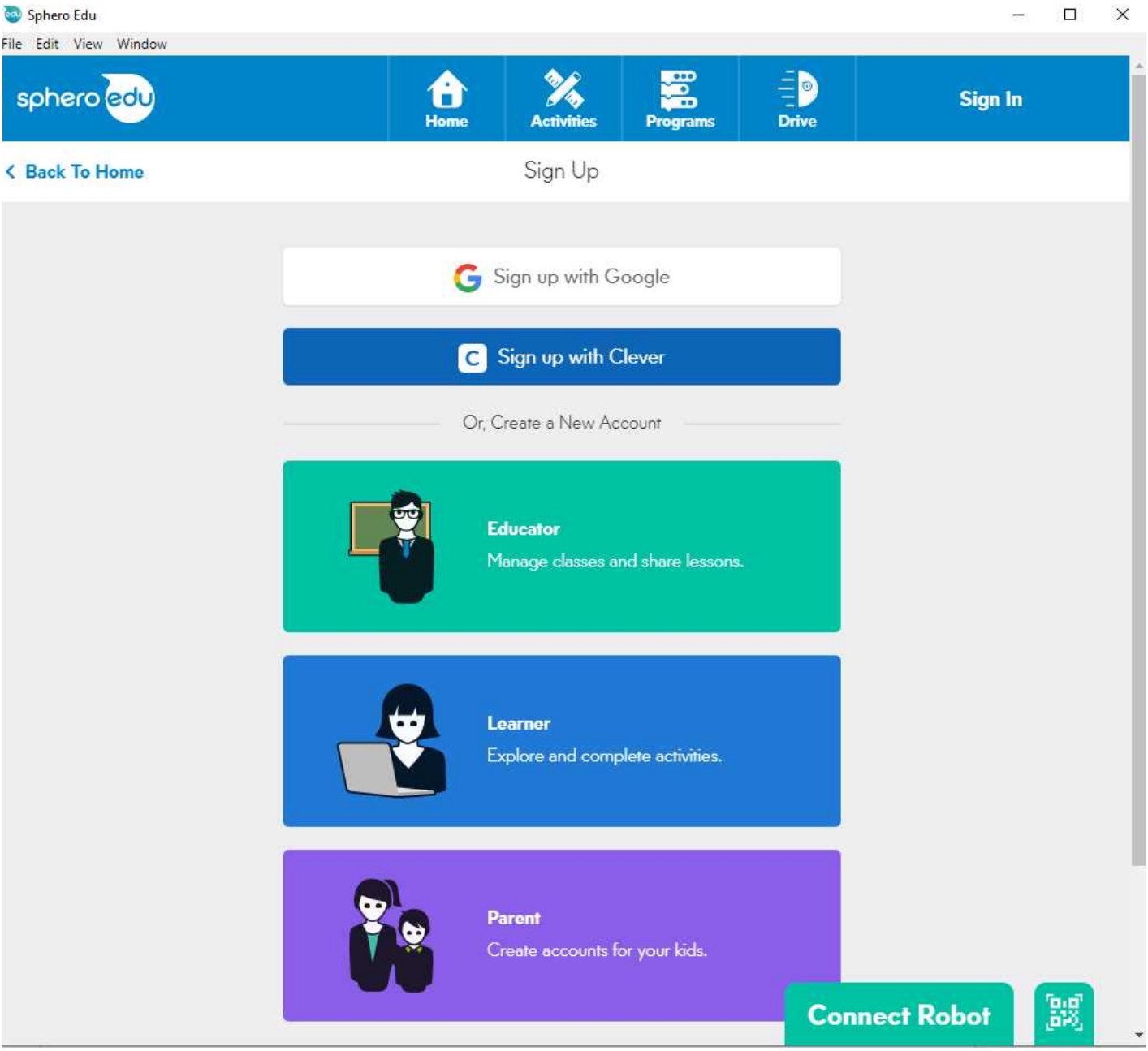
Para jugar, la versión móvil es mejor, por supuesto, pero... **para programar la versión escritorio** es la mejor opción, pues la pantalla pequeña es un incordio para poner los comandos, elegir, etc... programas cortos como Giro-mensaje o malabares se pueden hacer con el móvil, pero cuadrado es un sufrimiento la pantalla pequeña.

¡¡ HAY QUE REGISTRARSE !!!

Descargar el programa o la APP **NO NOS SIRVE PARA PROGRAMAR** tenemos que registrarnos



Nos pregunta por el perfil:



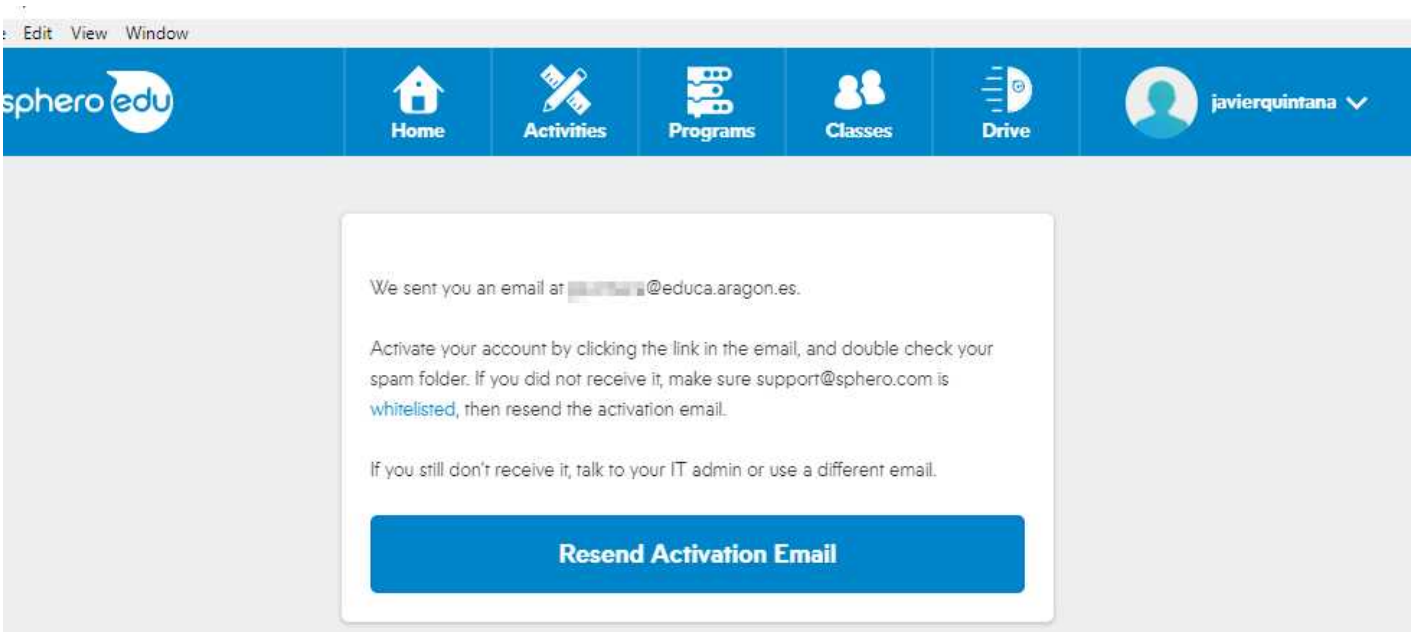
Una vez rellenado la ventana de registro **ya podemos CREAR PROGRAMAS**

A PARTIR DE AQUÍ, LOS SIGUIENTES PASOS NO SON NECESARIOS PARA REALIZAR ESTE CURSO

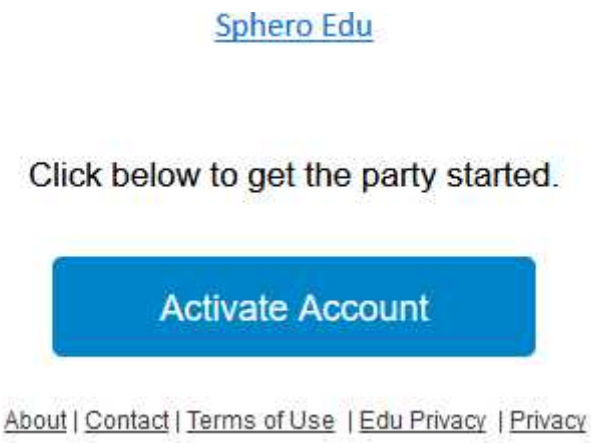
Registrarse pero con perfil de profesor, padre para crear clases o gestionar a

menores.

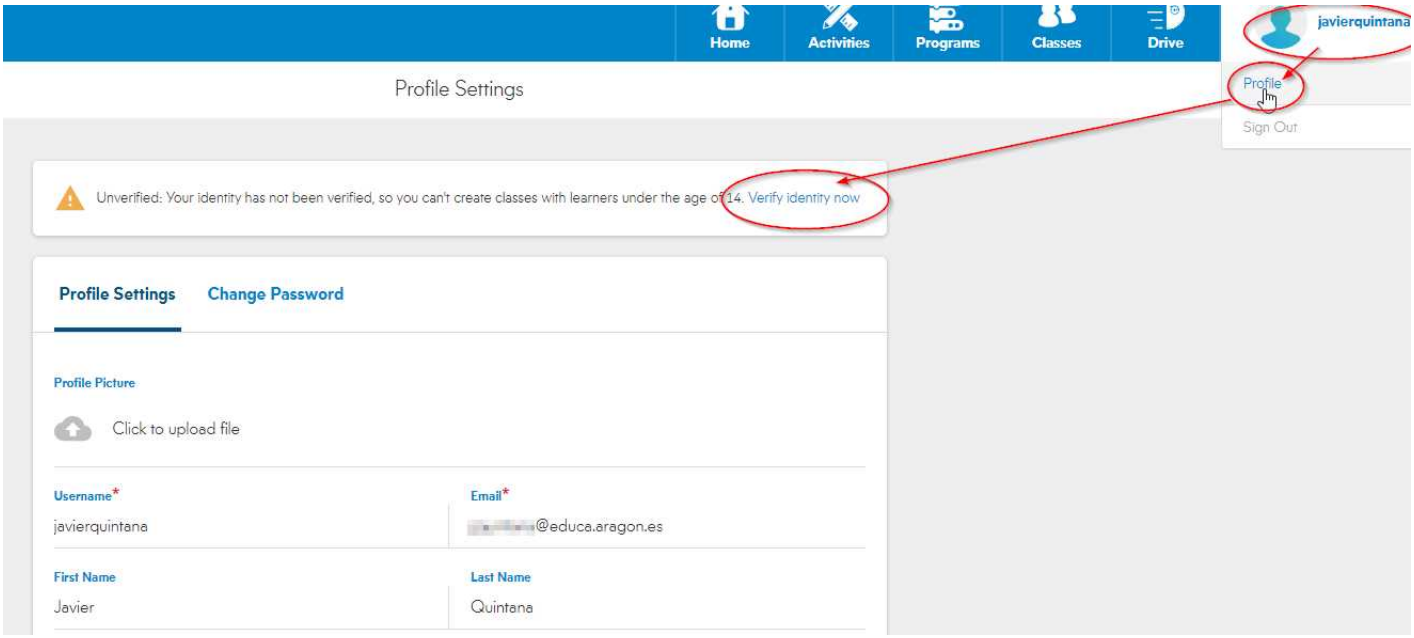
En este caso hay que pedir la confirmación por email



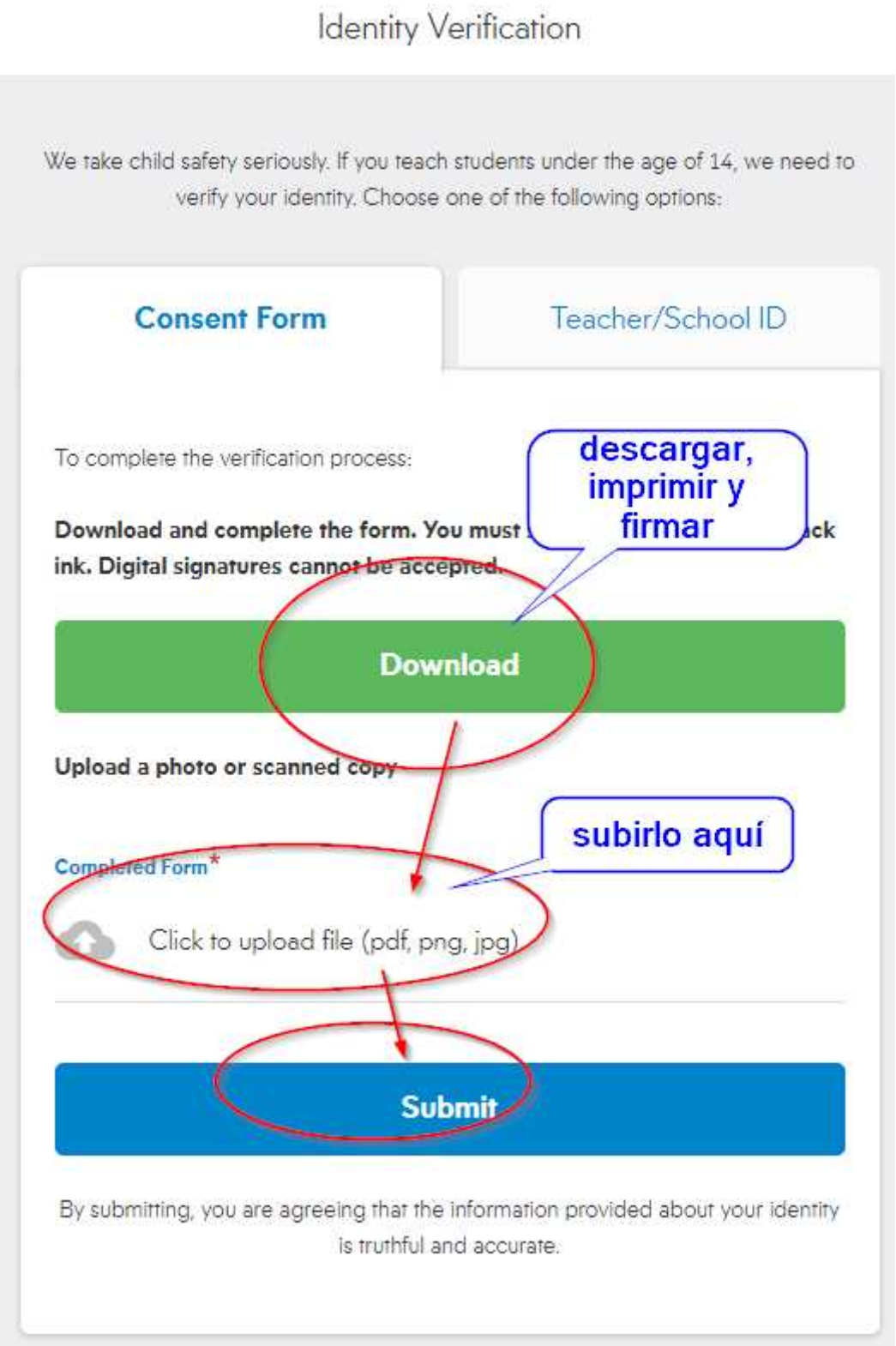
Se recibirá un email con este mensaje:



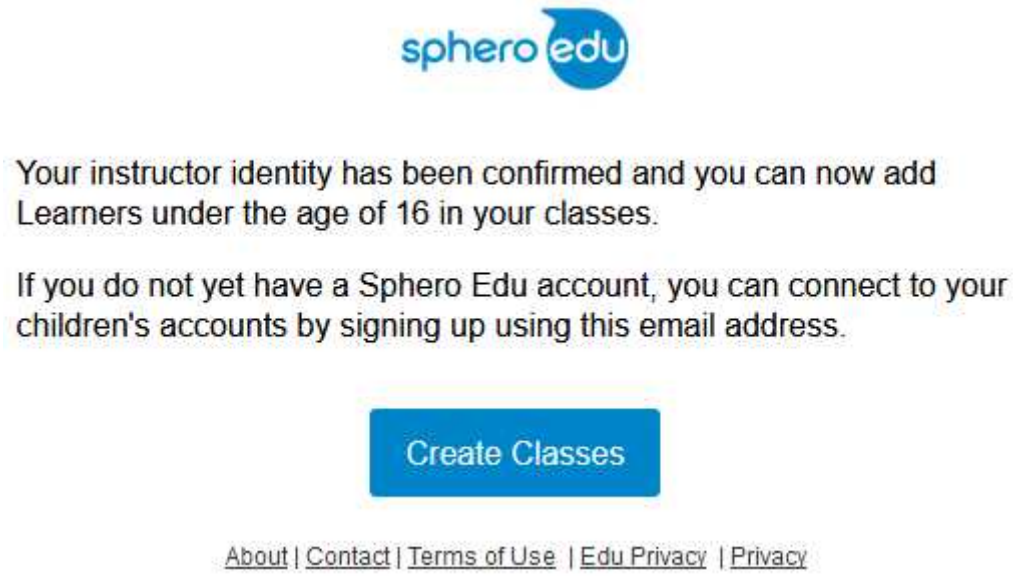
Si se pulsa, va a la página web de https://edu.sphero.com/cwists/category **EN TU PERFIL TIENES QUE FIRMAR TU CONSENTIMIENTO**



Esto implica **descargar la verificación, imprimirlo, firmar y enviar**



Al cabo de unas 24h te llega por email la confirmación:

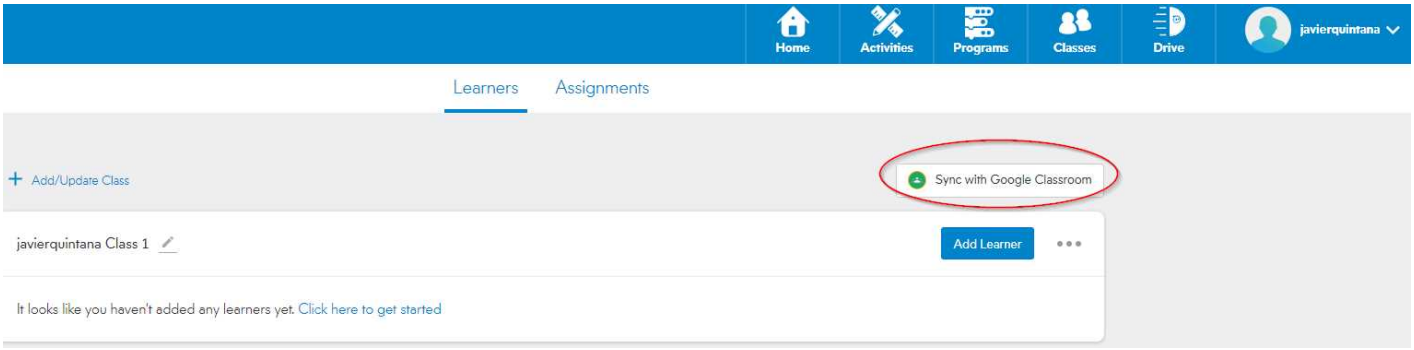


Con lo cual **ya puedes crear tu clase**

Crear una clase

Si nos hemos registrado correctamente (ver) nos saldrá una pantalla en https://edu.sphero.com con la posibilidad de crear una clase:

Puedes añadir a tus alumnos en el +Añadir clase de forma manual, pero nosotros recomendamos conectarlo con Google Classroom



En Actividades puedes diseñar tus actividades y asignarlas a tu clase de Google.

Como este curso es introductorio al manejo y programación de este robot, no entraremos en este aspecto.

1 Introducción

1.3 Encender

Lo primero que nos sorprende es que no hay botón de encender, sólo se puede **por software** y es necesario **un equipo con Bluetooth** ya sea un móvil o un portátil.

A Robot en standby

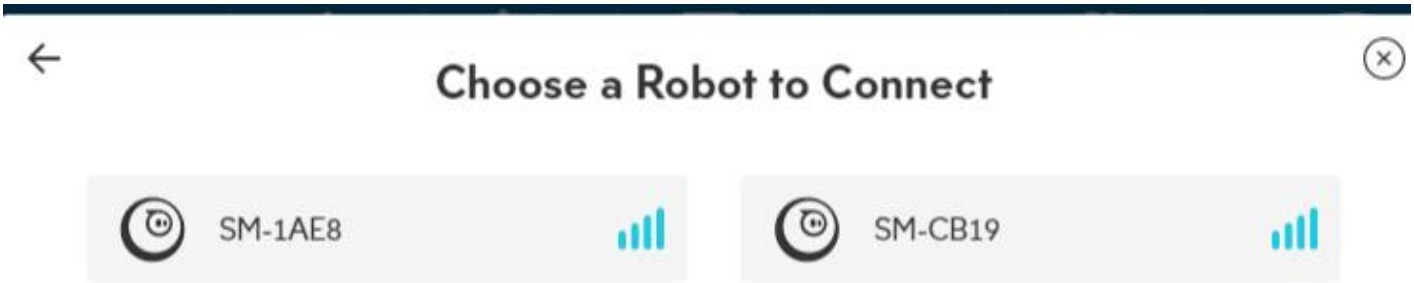
Verás en Apagar que si el robot no se apaga y no se utiliza, al cabo de un rato está en standby, para encenderlo hay que hacerlo por la aplicación, se queda en standby hasta que se agota la batería.

A1 Con el programa en Windows

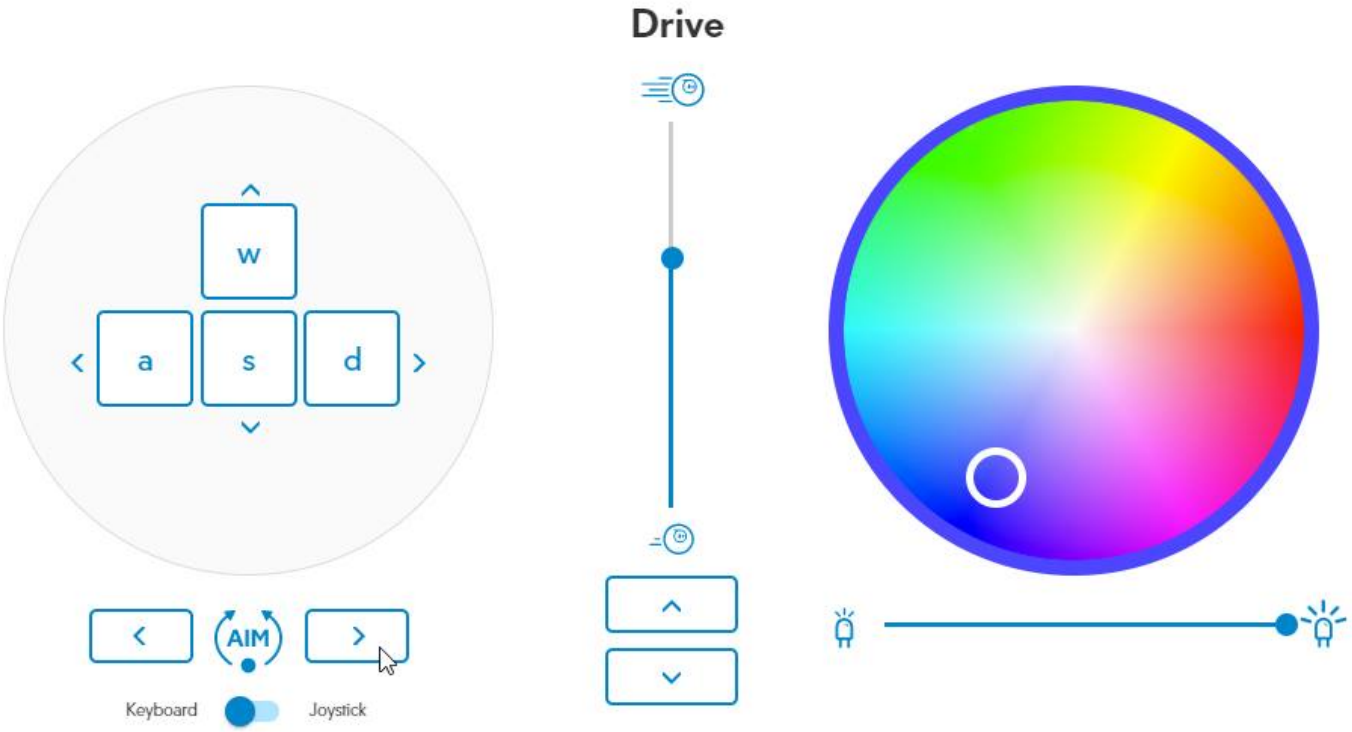
Entramos en el botón del robot:



No sale una lista de robots cercanos, y nos conectamos al que deseemos (en un grupo de alumnos, hay que tener claro cual es cual)

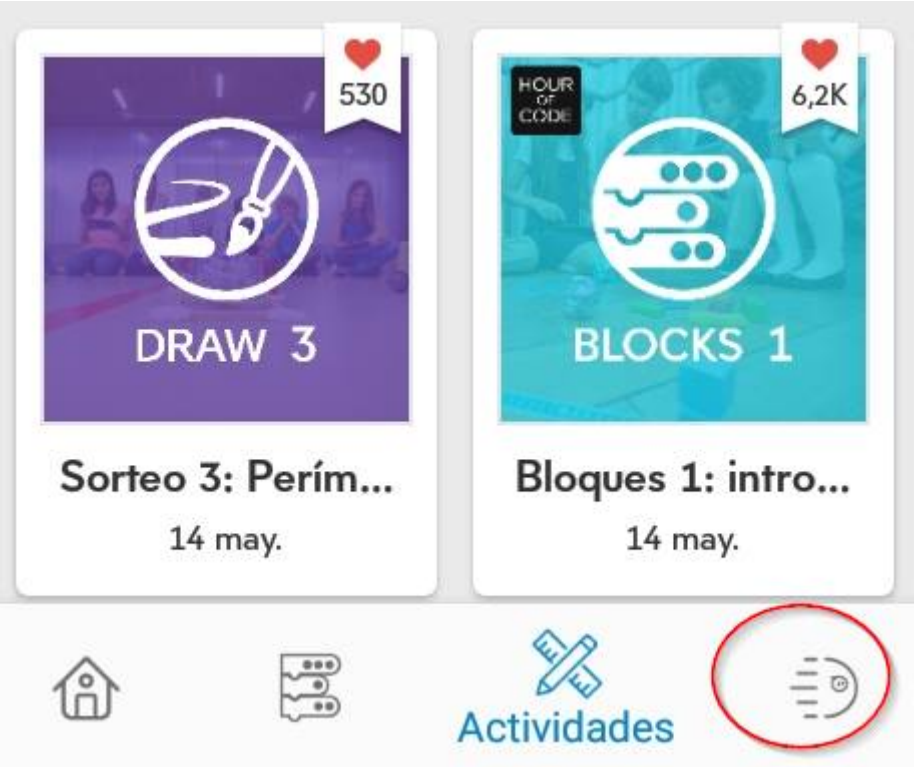


Y calibramos con los botones < y >, el punto luminoso del robot te tiene que ver a ti:



A2 Con la APP Android

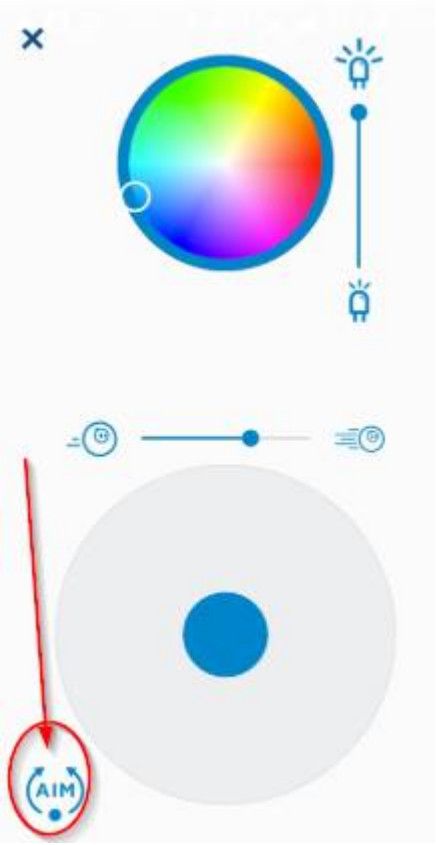
Buscamos el mismo botón del robot:



Nos conectamos con el robot:



Y calibramos



Ya sabes, poner el punto que te mire a ti:



B Robot totalmente apagado

En ese caso:

1.- Conectar el robot al cargador. 2.- Cuando esté lo suficientemente cargado (si se pone en **verde** es que está totalmente cargado, si está en **azúl** es que se está cargando) lo desconectas. 3.- Está entonces en modo standby: procede como en los pasos anteriores A.

Si el robot está totalmente cargado ¿tengo que también poner el cargador? SI, eso provoca el resetedo Bluetooth y modo standby.



Problemas en la conexión

Pues proceder igual que en B, conectar el cargador y quitarlo (si tiene suficiente carga), eso provoca el reseteo del Bluetooth.

En la misma APP te recomienda también hacer lo mismo

Copyright 2026


64 % 20:19
Conectar Sphero Mini
Para conectar, mantén tu dispositivo cerca de Sphero Mini
¿Todavía tienes problemas?
Más información

Copyright 2026

1 Introducción

1.4 Apagar

¿Dónde hay un botón de encendido y apagado? Respuesta no hay !!! esto tiene una desventaja: La batería **se descarga** al no poderlo apagar

Cuando dejas de utilizarlo, Sphero mini entra en modo stand-by que hace que tu batería se vaya gastando y cuando el día siguiente quieres usarlo: **ESTA DESCARGADO!!!** solución: Hay que apagarlo.

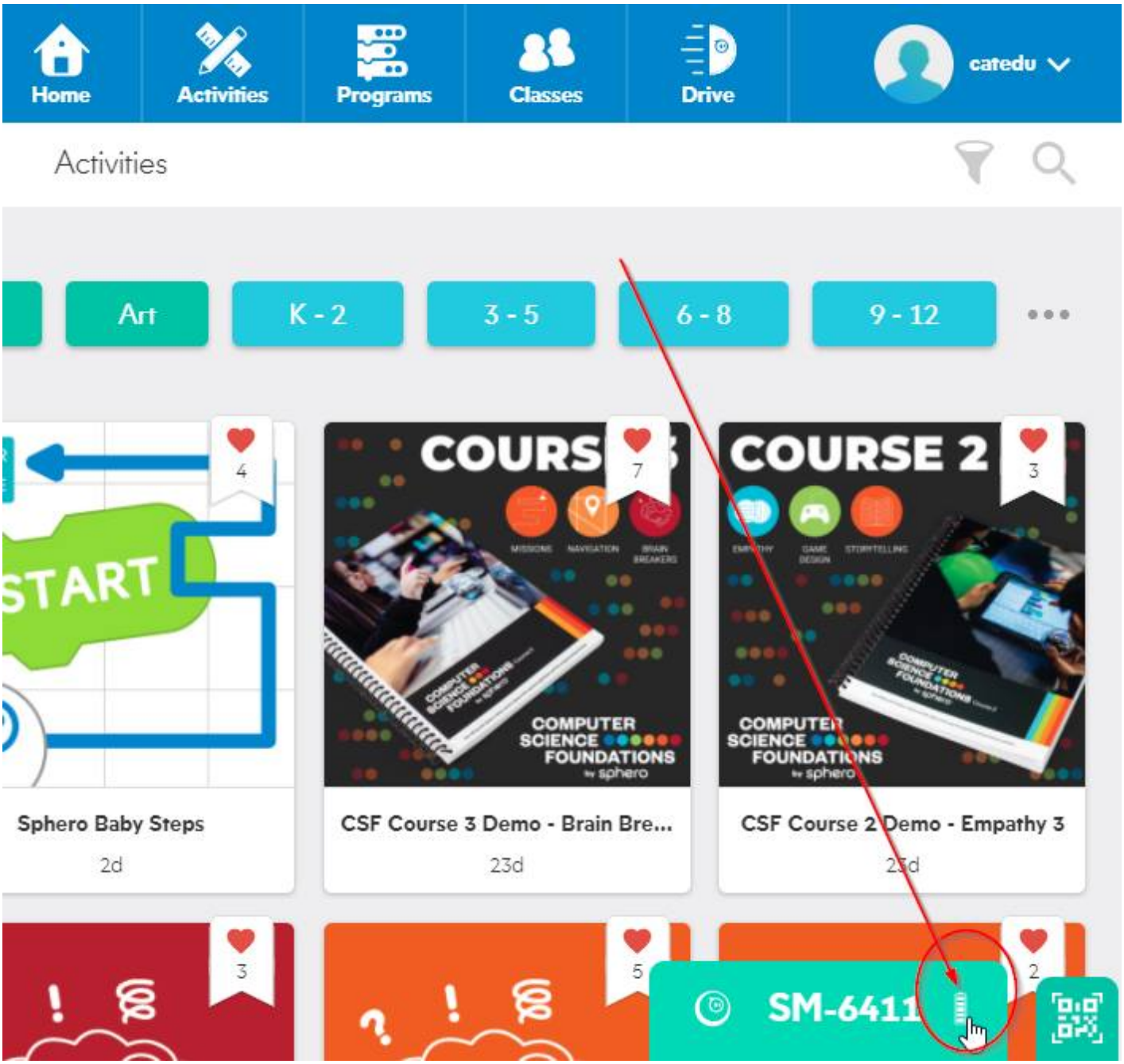
Para apagar el Sphero tienes que hacerlo POR SOFTWARE **UN ROLLO !!** pero hay que hacerlo

---

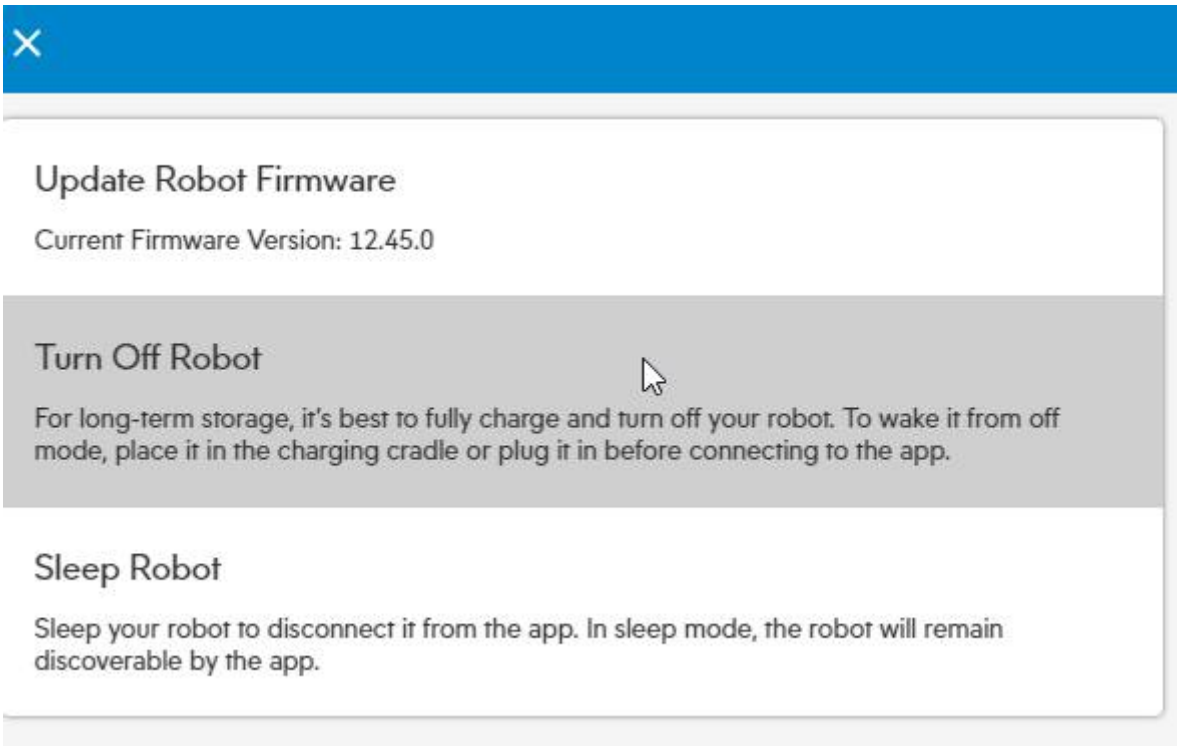
SI NO HACES ESTO ACORTAS LA DURACIÓN DE LA BATERÍA

Apagar desde el programa

Si lo haces desde el programa de Windows, (tiene que estar conectado el robot por supuesto), pulsas abajo en el robot:



Segunda opción:



Apagar desde la APP Shero Edu

Entramos en el robot (está en Inicio)

Copyright 2026

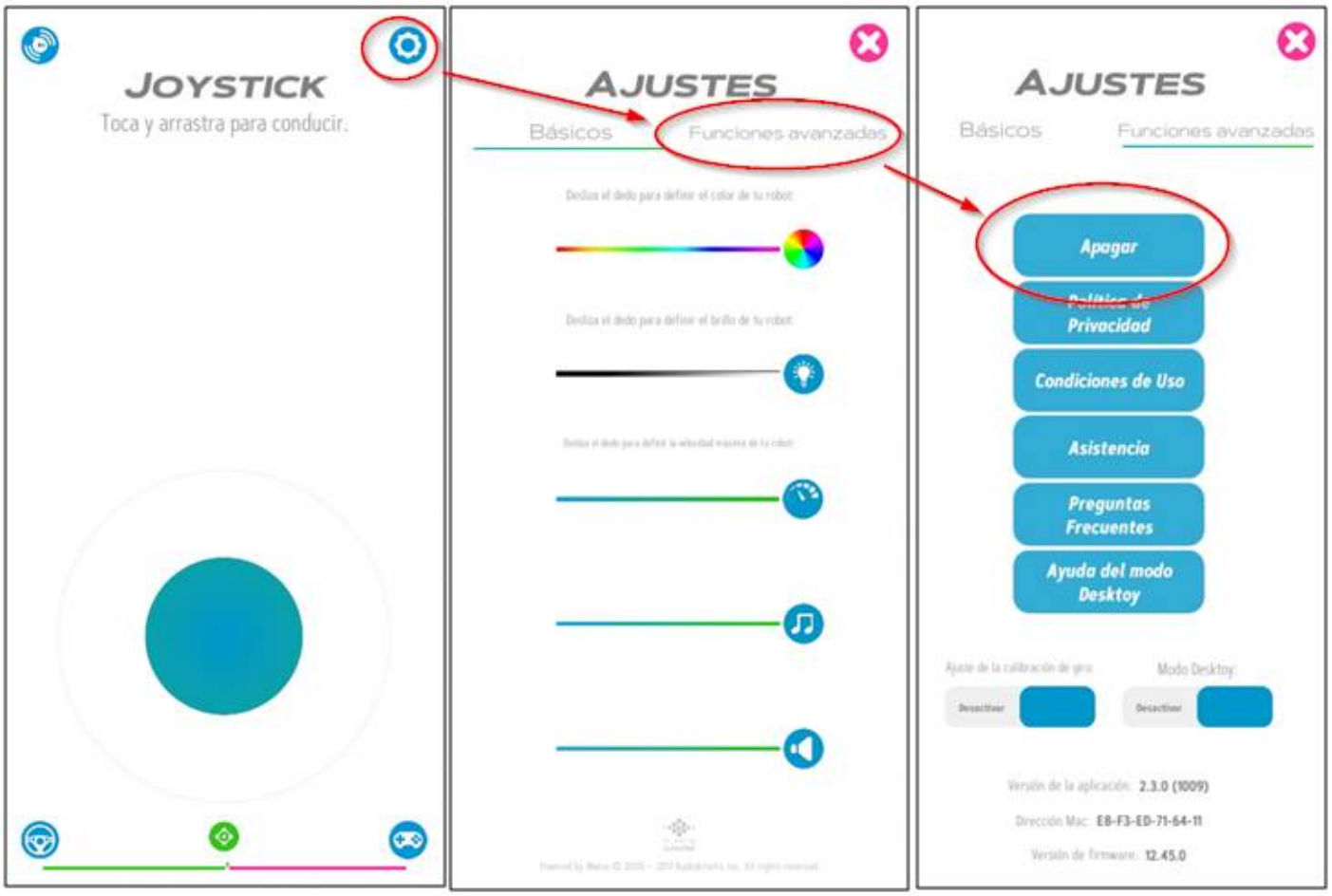

62 % 20:28
Modelos en 3D
Perfil
Ajustes
Iniciar sesión
SM-CB19
58729074
13 h
58729074 posted the program
color gaurd.
Inicio



Copyright 2026

Apagar desde la APP Sphero Play

Entra en la APPy dar a configuración -> opciones avanzadas -> Apagar



2 A programar!!

Copyright 2026

2 A programar!!

2.1 Giro-mensaje

Objetivo

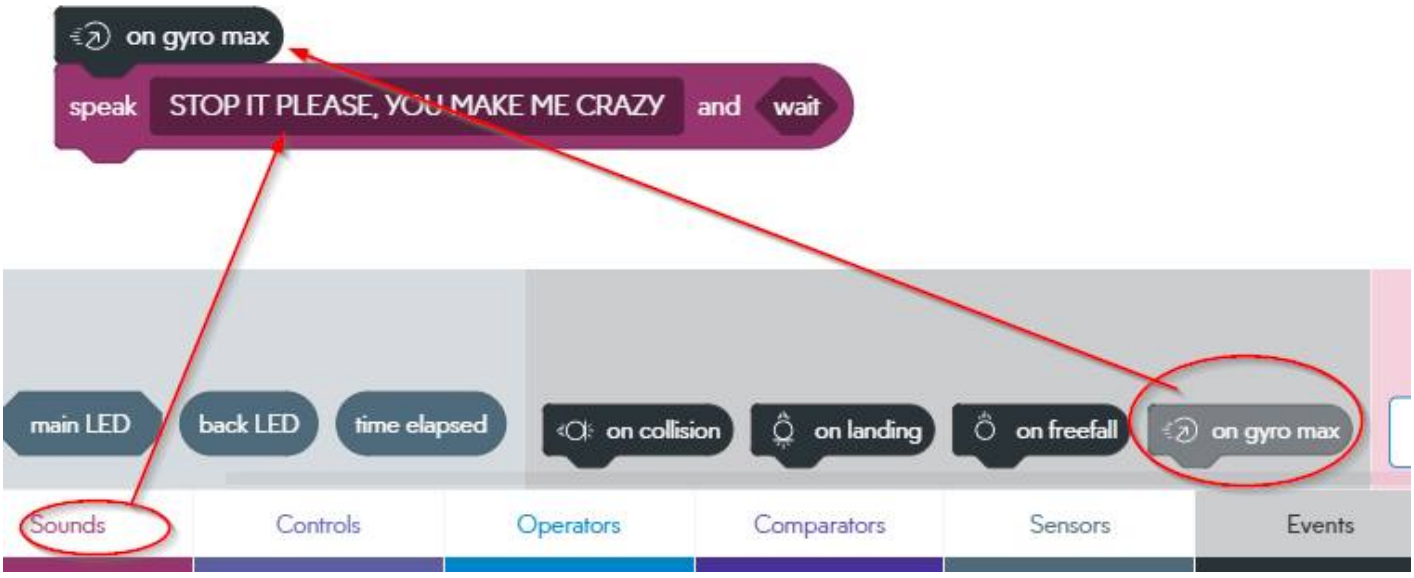
Un programa sencillo que si giro el Sphero Mini diga un mensaje

Programa

Cogeremos el evento GiroMax y le añadimos el audio con un mensaje

El programa lo puedes encontrar en este enlace

https://edu.sphero.com/remixes/5451216



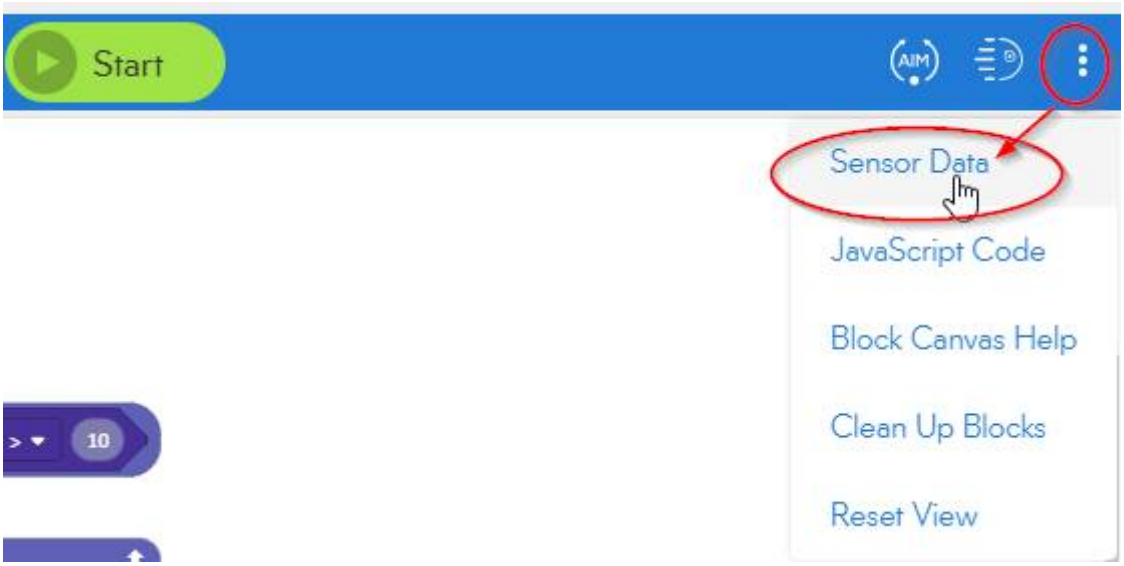
Resultado

https://www.youtube.com/embed/xfpNphAzxDo

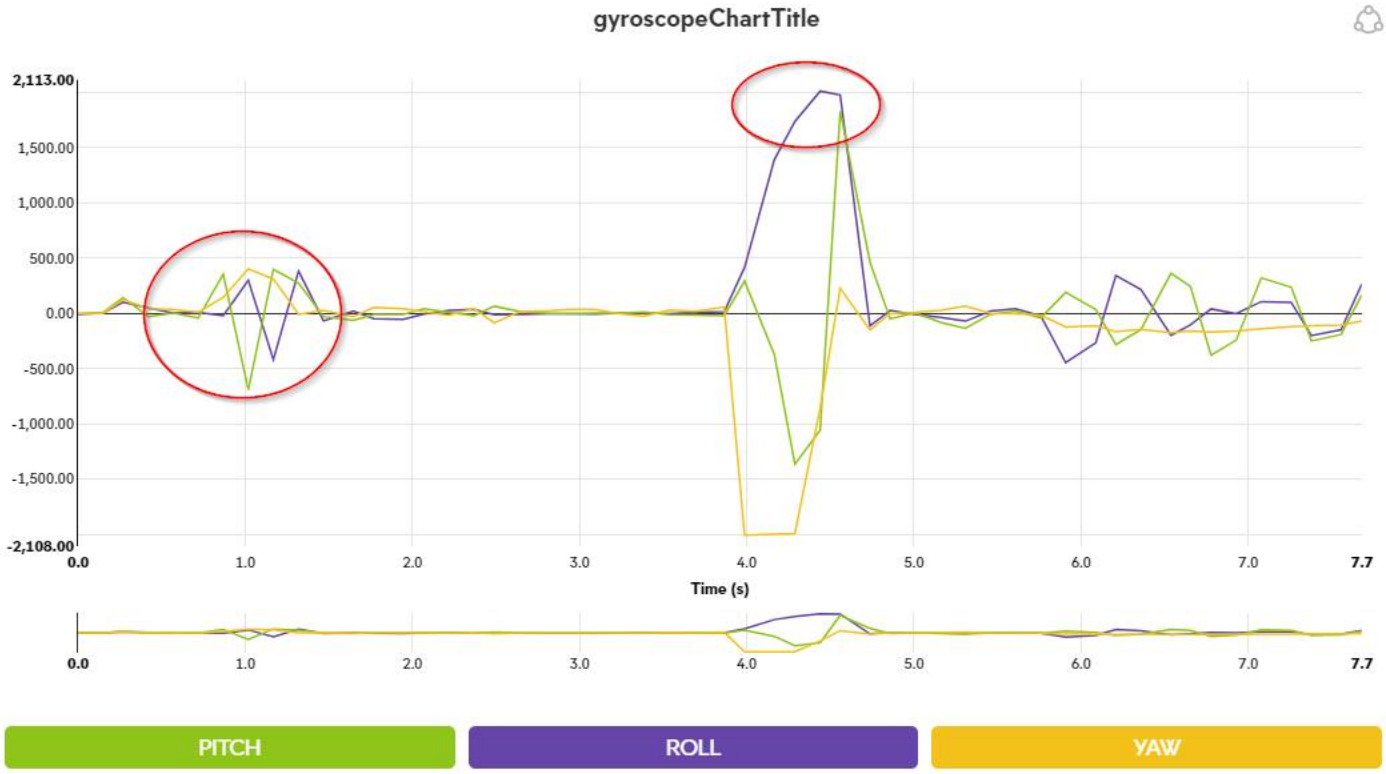
¿ Problemas ?

Puede ser que tu Sphero-mini no llegue al giro máximo, o no quieres que haya que ser tan bruto para que salte el mensaje

En Sensor Data puedes ver qué cantidad de giro ha hecho



si queremos que salte ya en el primer caso, sin necesidad de llegar al máximo:



Simplemente bajamos la sensibilidad con otro tipo de programa, sin utilizar el evento giro máximo. En la ilustración el programa modificado para una sensibilidad de 100:



Elegimos la rotación horizontal pues es la que se va a utilizar:



... pero ¿da igual el sentido de rotación? R: No, el sentido de rotación de las agujas del reloj da lugar a valores negativos.

2 A programar!!

2.2 Malabares

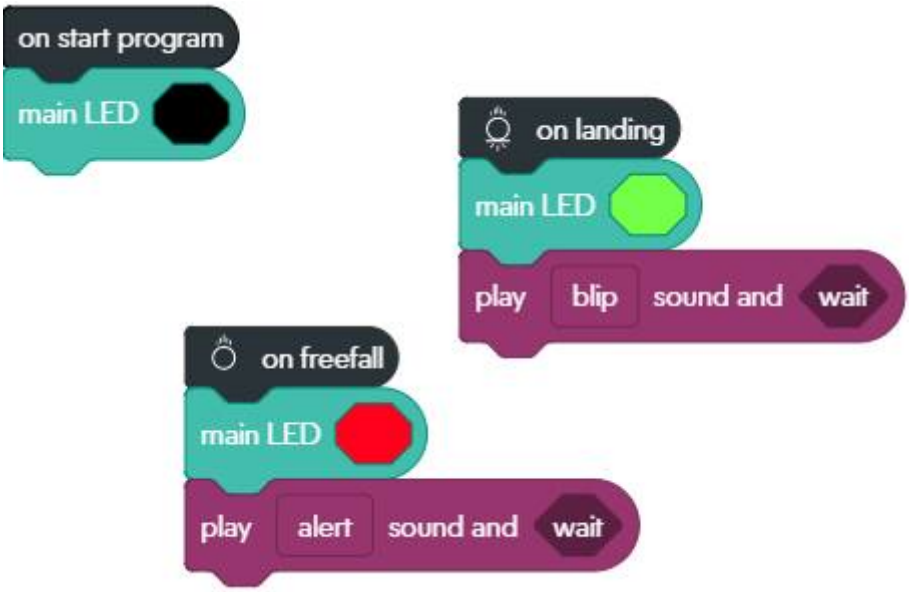
Objetivo

Un programa que al lanzar el mini emita un sonido y al aterrizar otro efecto de sonido-luminoso

Programa

Cogeremos los eventos de "caída libre" y "aterrizaje" haremos unos efectos distintos en cada caso. Son eventos que responden al sensor de aceleración en el eje vertical (en caída libre tendrá un valor nulo después de pasar por un máximo y en aterrizaje será un valor máximo después de pasar por un valor nulo) El programa lo puedes encontrar en este enlace

https://edu.sphero.com/remixes/5451292



Resultado

https://www.youtube.com/embed/Wo6s6GsECEs

¿Por qué a veces no se sincroniza el movimiento con el sonido?

Si nos fijamos, dependiendo de nuestro equipo (ordenador o móvil) hay un retraso en la comunicación Bluetooth, que hace que el efecto no está sincronizado con el evento.

2 A programar!!

2.3 Cuadrado

Objetivo

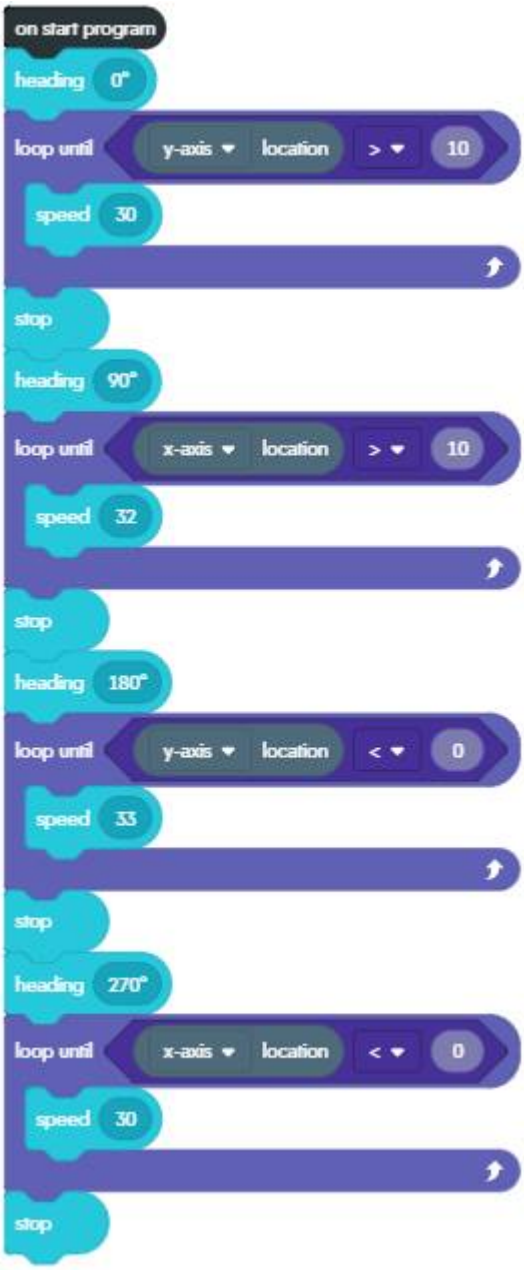
Un programa que mueva a Sphero Mini en un cuadrado 10cm x 10 cm y visualizaremos el resultado en el registro de sensores del mismo programa.

Programa

Este programa el truco está en hacer un bucle que mueva el robot hasta que la posición de la ordenada sea 10 cm, orientamos el sphero-mini y a por el siguiente lado. Total 4 bucles.

El programa lo puedes encontrar en este enlace

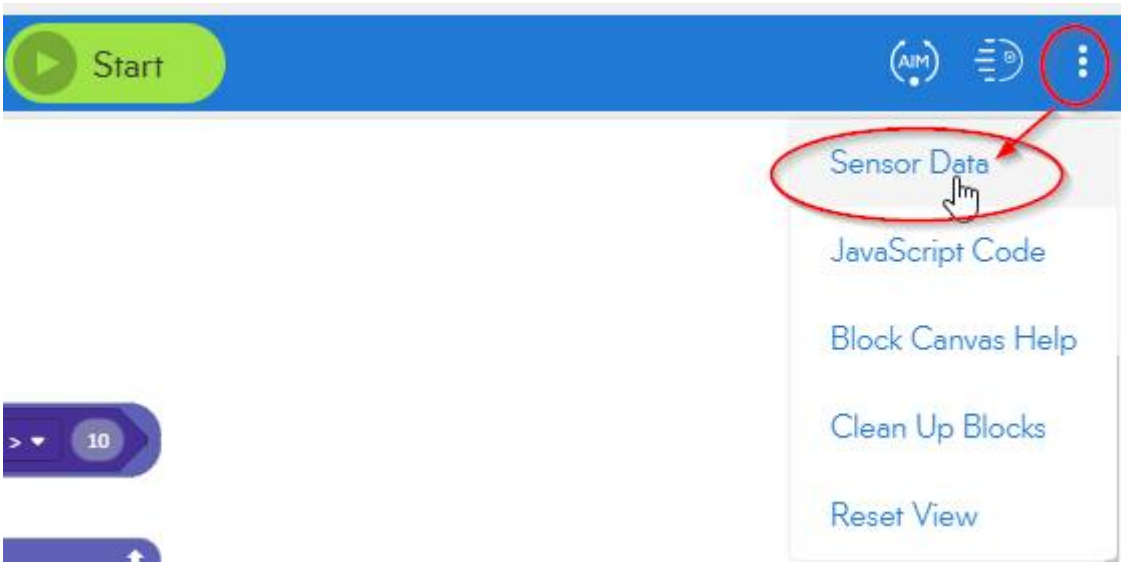
https://edu.sphero.com/remixes/5450219



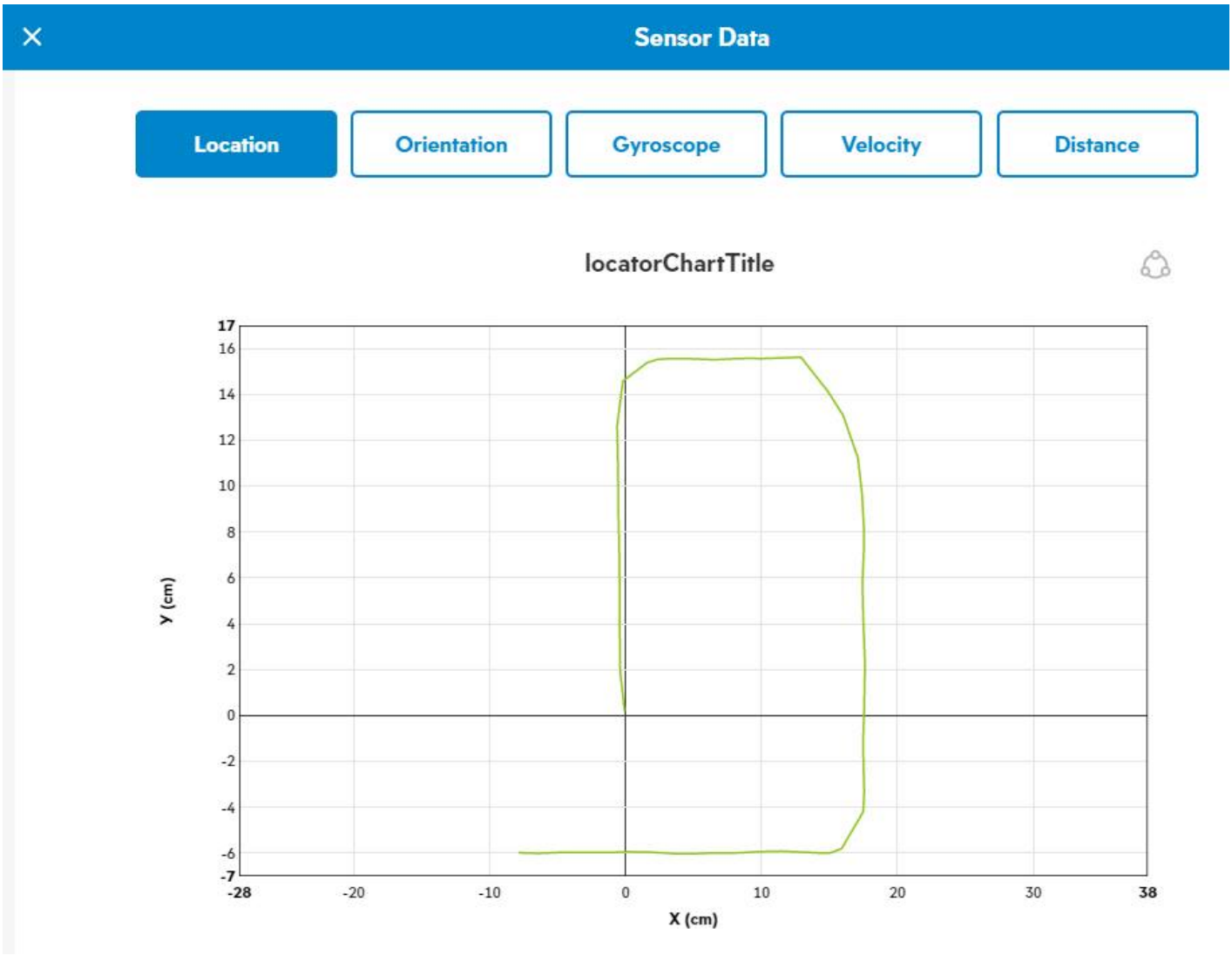
Sensor data

Una vez ejecutado, podemos ir al registro de los sensores:

Copyright 2026



y la verdad es que del diseño 10cm x 10cm, ha salido un cuadrado un poco .... vamos que el error es casi del 70%



Pero **los sensores SI que miden bien**, lo puedes ver en el vídeo:

Resultado

https://www.youtube.com/embed/YxkXVzXxxsE

¿Por qué es tan impreciso?

Ya te dimos una pista en Giro-mensaje: La comunicación entre la aplicación y el robot, cuando la aplicación manda el mensaje de que gire y haga el otro lado, ese retardo él ya ha recorrido 7cm.

Copyright 2026

2 A programar!!

2.4 ¿Ganaré la lotería?

Objetivo

Un programa sencillo de adivinanza

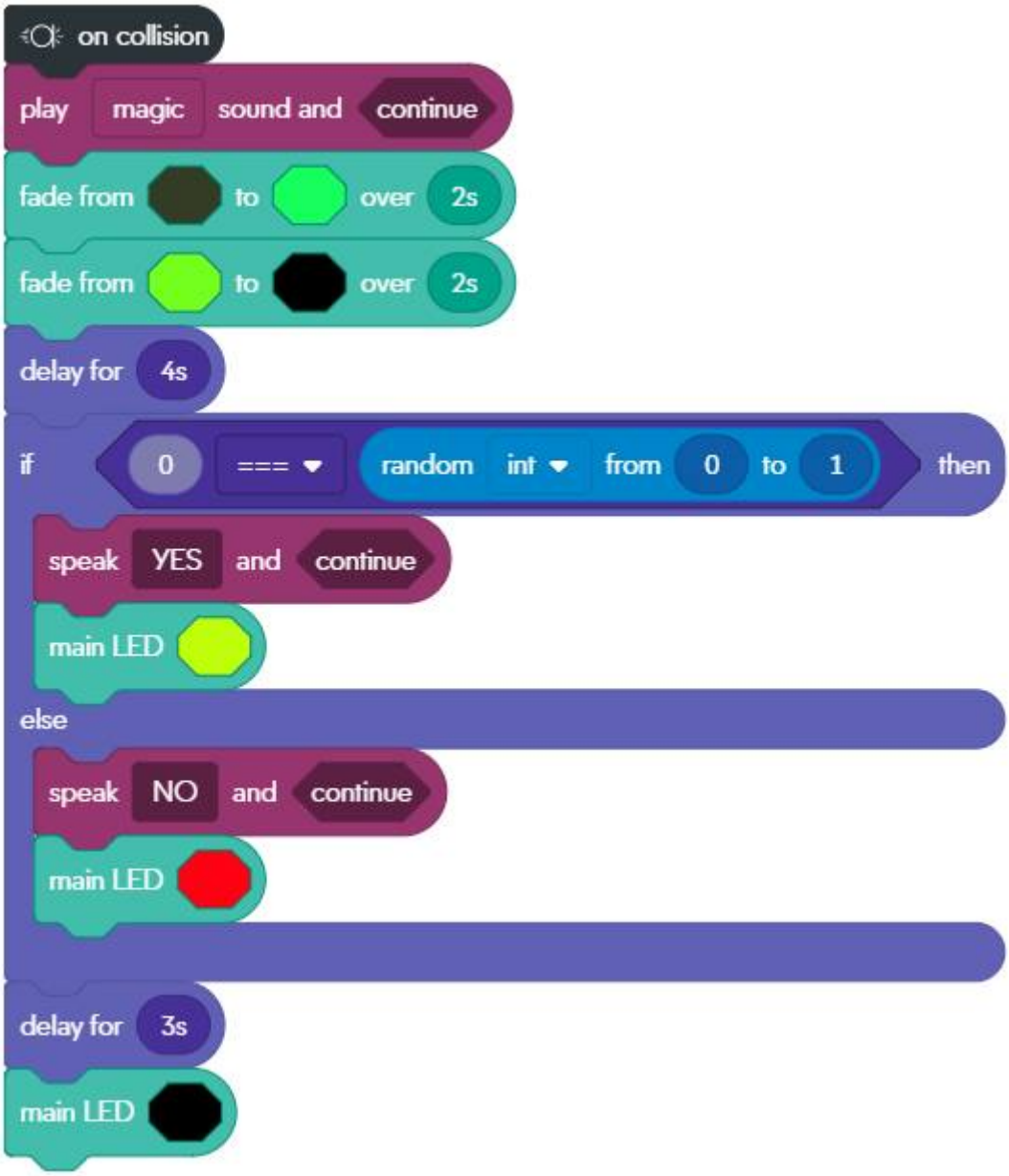
Programa

Cogeremos el evento y con el evento colisión le añadimos efectos y audio. Genearemos una variable aleatoria que nos dirá si ganaremos o no la lotería.

El programa lo puedes encontrar en este enlace

https://edu.sphero.com/remixes/5435323

Copyright 2026



Resultado

https://www.youtube.com/embed/QiQP80wh7N4

Copyright 2026

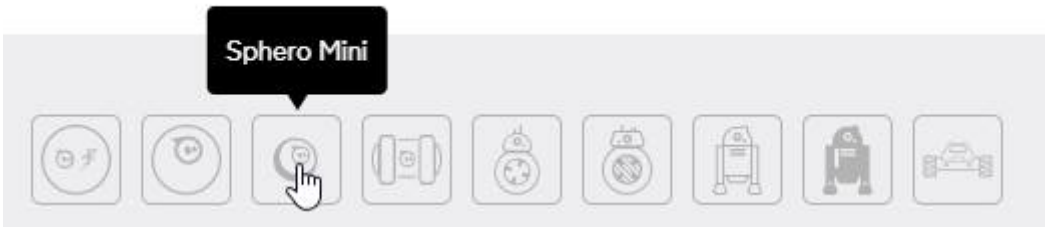
2 A programar!!

2.5 Actividades de otros

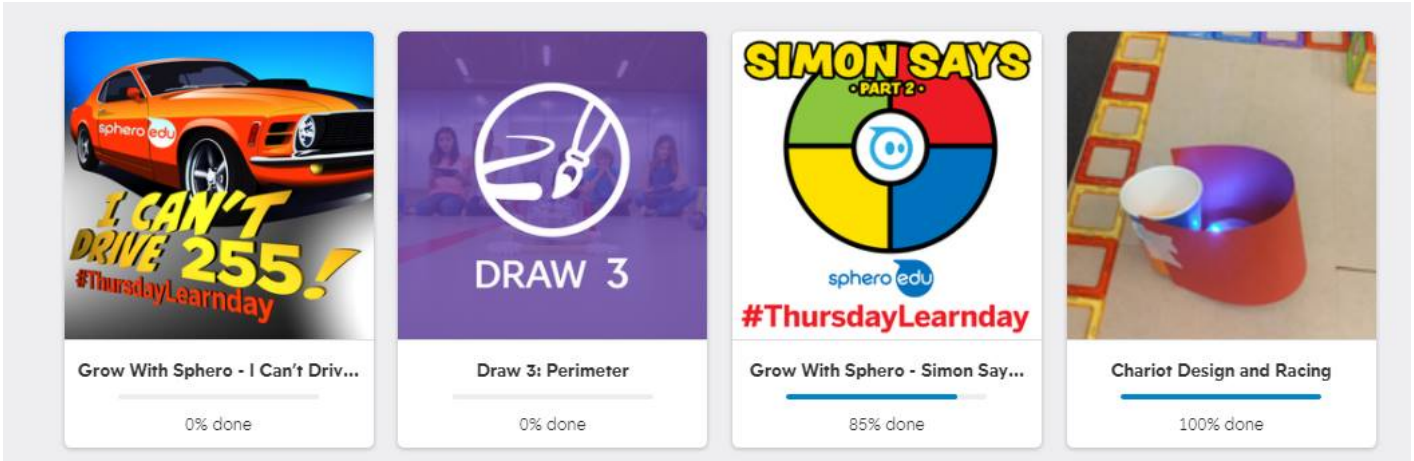
Desde el programa, lo primero que nos encontramos son actividades públicas creadas por terceros. Ante tanta variedad, lo mejor es filtrar los contenidos:



Filtramos por SHERO MINI



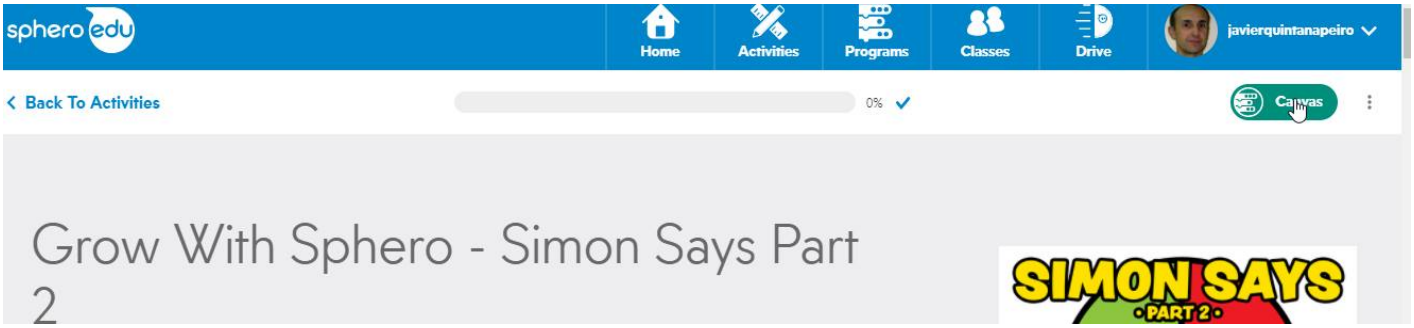
Entramos en la que nos interese:



Empezamos la actividad



Entramos en el **canvas**



Y se abre una ventana flexible donde programar en la parte derecha y ver los pasos tutorizados a la izquierda.

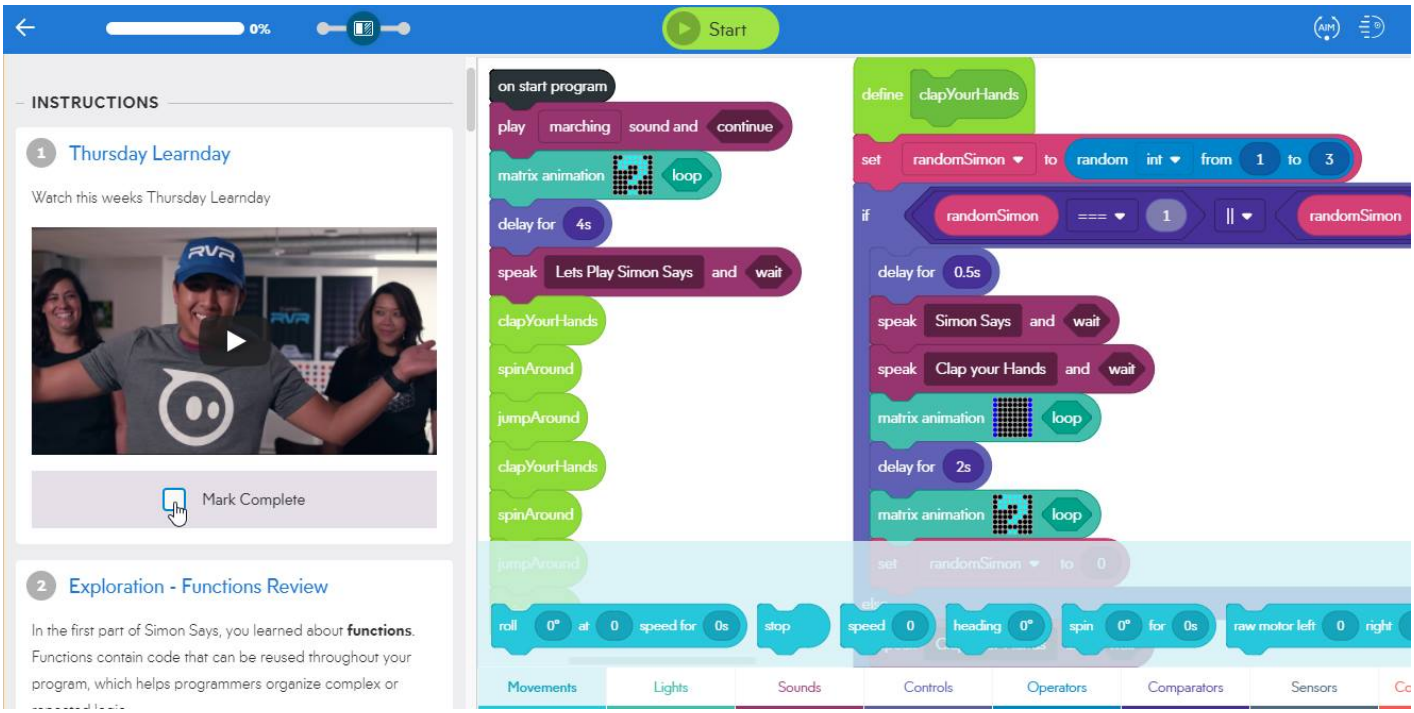
Barra de desplazamiento principal

Existe un scroll en la barra azul de arriba con 3 posiciones: * maximizar izquierda * las dos ventanas * maximizar la derecha

En la siguiente ilustración el scroll está enmedio, y en la ventana de creación ya nos da una propuesta de programa. (muchas veces nos encontramos que no hay propuesta, que tienes que hacer tú el programa).

También nos podemos encontrar fallos, como en este caso el programa no está preparado para **Shpero mini**, pues la instrucción *matrix animation* es para la versión no mini, hay que quitarla.

Copyright 2026


0%
Start
INSTRUCTIONS
1 Thursday Learnday
Watch this weeks Thursday Learnday
Mark Complete
2 Exploration - Functions Review
In the first part of Simon Says, you learned about functions. Functions contain code that can be reused throughout your program, which helps programmers organize complex or
on start program
play marching sound and continue
matrix animation loop
delay for 4s
speak Lets Play Simon Says and wait
clapYourHands
spinAround
jumpAround
clapYourHands
spinAround
define clapYourHands
set randomSimon to random int from 1 to 3
if randomSimon === 1 || randomSimon
delay for 0.5s
speak Simon Says and wait
speak Clap your Hands and wait
matrix animation loop
delay for 2s
matrix animation loop
roll 0° at 0 speed for 0s
stop
speed 0
heading 0°
spin 0° for 0s
raw motor left 0 right
Movements
Lights
Sounds
Controls
Operators
Comparators
Sensors

2 A programar!!

2.6 Conclusiones

Una vez visto por encima este robot, queremos comentar las principales ventajas/inconvenientes que vemos en este robot. Son opiniones nuestras que valoramos desde CATEDU y son totalmente criticables y perfectamente puedes no estar acuerdo o ponerte en contacto con nosotros si ves que tendríamos que cambiar algo:

BATERÍA: PUNTO CRÍTICO DE ESTE ROBOT

Que se haga por software el apagado es un gran inconveniente, pues si no se hace (y sospechamos que por comodidad no se hará) PERJUDICA A LA VIDA ÚTIL DE LA BATERÍA ¿qué les costaba poner un micro-interruptor? y además no es fácil su sustitución.

SENSOR DATA NO VALE PARA EXPERIMENTOS DE CINEMÁTICA.

El registro de posición, velocidad y giro es tentador para hacer experimentos de dinámica-cinemática STEAM con el robot pero ... EL SENSOR DE POSICIÓN Y VELOCIDAD SE REALIZA POR EL MOTOR PASO A PASO INTERNO DEL ROBOT ¿qué quiere decir esto? pues que mide la distancia y velocidad cuando **hacemos mover el robot por software** pues es casi imposible que el robot *ruede internamente* por impulso externo.

Prueba el siguiente ejemplo: lanzar Sphero por una superficie. Vemos que el sensor detecta sólo que lo hemos movido menos de 2 cm. ¿Por qué? porque **no ha rodado internamente** excepto al final en el breve frenado:

https://www.youtube.com/embed/3d1ckVvrsIc

Este es el resultado:



FALTA PROGRAMACIÓN BIDIRECCIONAL EN EVENTOS: APP SPHERO EDU ↔ ROBOT SPHERO MINI

El robot se comunica con la app (con algo de retraso por el Bluetooth) envía los datos de los sensores... luego hay comunicación entre Sphero-mini y la APP pero **sólo en un sentido Eventos robot Sphero-Mini ➡ Aplicación** ¿Por qué no existe el otro sentido Sphero-mini ⬅ Eventos en el disipositivo de la Aplicación?

Por ejemplo en Sphero Play hay juegos que utilizan el móvil como joystick, pero en Sphero Edu no podemos programar enviar órdenes **según eventos en el dispositivo donde está SheroEdu** a

Sphero-mini, por ejemplo usar las teclas del teclado como joystick.

ACTIVIDADES DE OTROS

La gran diversidad de actividades que se publican produce un efecto de *infoxicación* y encontramos muchas actividades con poco interés STEM. No obstante si ves algo interesante, puedes ayudarnos y publicarlo en el muro.

Lo que sí que nos gusta

- Ocupa poco
- Buen diseño
- Es muy resistente a golpes.

3 Para saber más ...

Copyright 2026

3 Para saber más ...

3.2 Muro

Pon si quieres poner ejemplos tuyos o de otros interesantes:

https://padlet.com/embed/rr8y5ryo4r8q

Hecho con Padlet

Copyright 2026

3 Para saber más ...

Créditos

Autoría

- {{ book.authors[0] }}

Cualquier observación o detección de error en soporte.catedu.es

Los contenidos se distribuyen bajo licencia **Creative Commons** tipo **BY-NC-SA** excepto en los párrafos que se indique lo contrario.