Enseña Pensamiento Computacional Con Lego Wedo

En este curso vamos a dar rienda a la imaginación LEGO WEDO es una propuesta donde la construcción e imaginación es una parte importante, pero sin olvidar del pensamiento computacional. Los algoritmo...

Introducción

En este curso vamos a dar rienda a la imaginación

LEGO WEDO es una propuesta donde la construcción e imaginación es una parte importante, pero sin olvidar del pensamiento computacional.

Los algoritmos son sencillos, sólo tenemos un motor como salida, y dos sensores, pero la creatividad es muy elevada.

1. ¿Qué es LEGO WEDO?
2. Construcciones propuestas
3. La creatividad

ATENCION: El curso (y su préstamo) se basan en LEGO WEDO 1 QUE ESTA DESCATALOGADO (2018 sólo se vende LEGO WEDO 2) pero las destrezas que se aprenden con LEGO WEDO 1 son útiles para aprender LEGO WEDO 2.

Aquí se pueden ver las construcciones de LEGO WEDO2

CENTRO ARAGONÉS de TECNOLOGÍAS para la EDUCACIÓN

CATEDU

Un poco de orden... el pensamiento computacional

¿Esto es una moda?

No sabemos qué futuro van a encontrar nuestros alumnos, pero sí que sabemos que por ejemplo el **Inglés** será importante en su entorno futuro. Pues igual con las TIC, no es una moda, hace tiempo que está, y seguirá. **El pensamiento computacional es el idioma de los ordenadores.**

Vale, y … este curso ¿Dónde se encuadra?¿para qué edad es recomendada?

Buena pregunta... para enseñar el pensamiento computacional tenemos dos caminos, totalmente compatibles:

- **La programación**, que sería como enseñar un nuevo idioma.
- **La robótica** que sería como practicar este idioma con un nativo, luego antes hay que saber el idioma.

En CATEDU hemos elaborado esta **hoja de ruta** de herramientas y edades, hay otras herramientas y otros criterios TOTALMENTE VALIDOS, este es el nuestro, lo que hemos elegido en los cursos de Aularagon y que enseñamos aquí como orientación, pero no se debe de tomar al pie de la letra.

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

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.

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 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

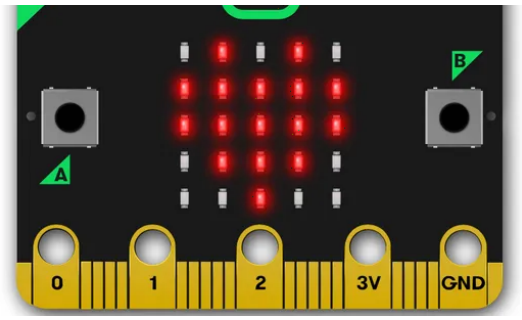
https://player.vimeo.com/video/18390711?h=b5844e7753

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

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 su uso está extendido entre 25 millones 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.

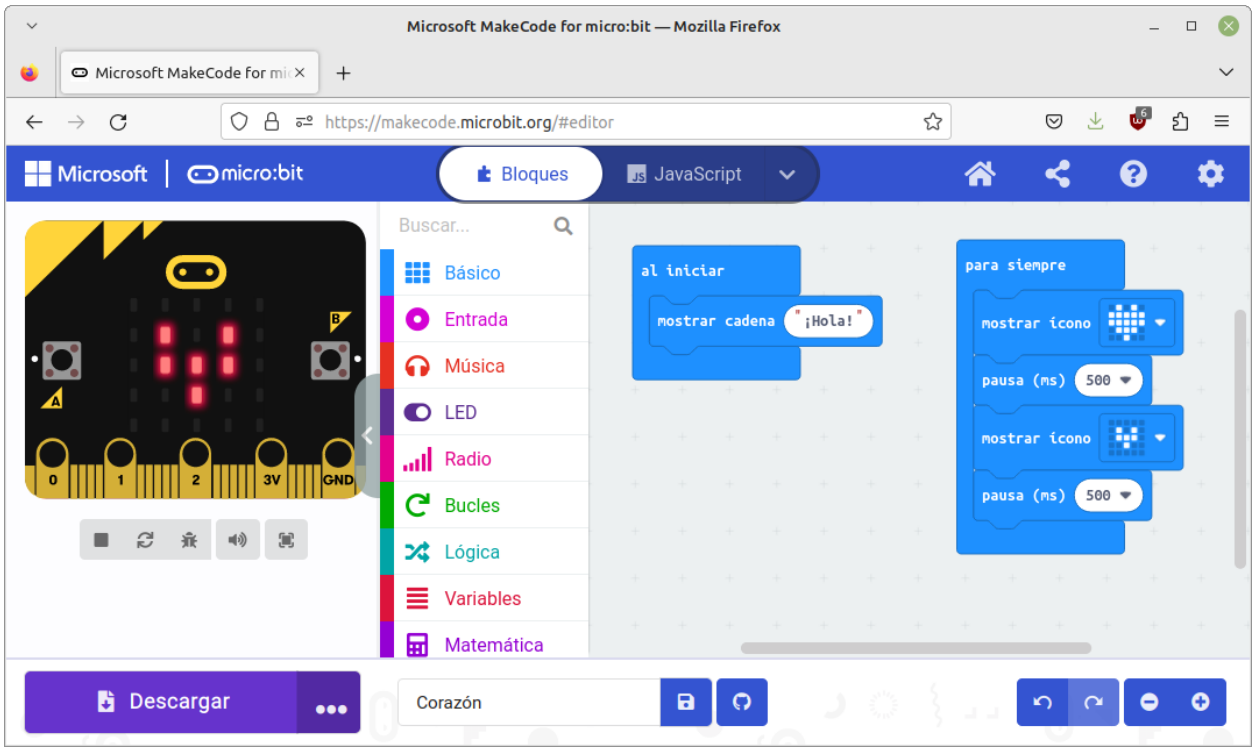
- La colaboración con entidades asociadas que compartan una misma visión para ofrecer programas educativos de alto impacto en todo el mundo.

> 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.

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.

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.

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.

1. ¿Qué es Lego Wedo?

1. ¿Qué es Lego Wedo?

1.1 Introducción

<

CENTRO ARAGONÉS de TECNOLOGÍAS para la EDUCACIÓN
CATEDU

1.3 ¿Qué es LEGO WeDo?

Es un set de LEGO® que permite la construcción de 12 diferentes actividades (y las que permita la imaginación) para introducir al alumno en robótica

En este curso nos centramos en el** LEGO WEDO 1.0 modelo 9580 [aprox 120€](http://www.robotix.es/es/lego-y-education-wedo-construction-set-lego-education)**

- **Se me ha perdido alguna pieza ¿dónde puedo encontrar recambios?**
  - Piezas comunes [aprox. 5€](http://www.robotix.es/es/le-replacement-pack-le-wedo-1-lego-education) - Sensor distancia [aprox 20€](http://www.robotix.es/es/motion-sensor-movimiento-lego-education) - Sensor inclinación [aprox 20€](http://www.robotix.es/es/tilt-sensor-inclinacion-lego-education) - Motor [aprox 20€](http://www.robotix.es/es/medium-motor-wedo-20-lego-education) - HUB USB [aprox 30€](http://www.robotix.es/es/lego-usb-hub-lego-education)

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

1. ¿Qué es Lego Wedo?

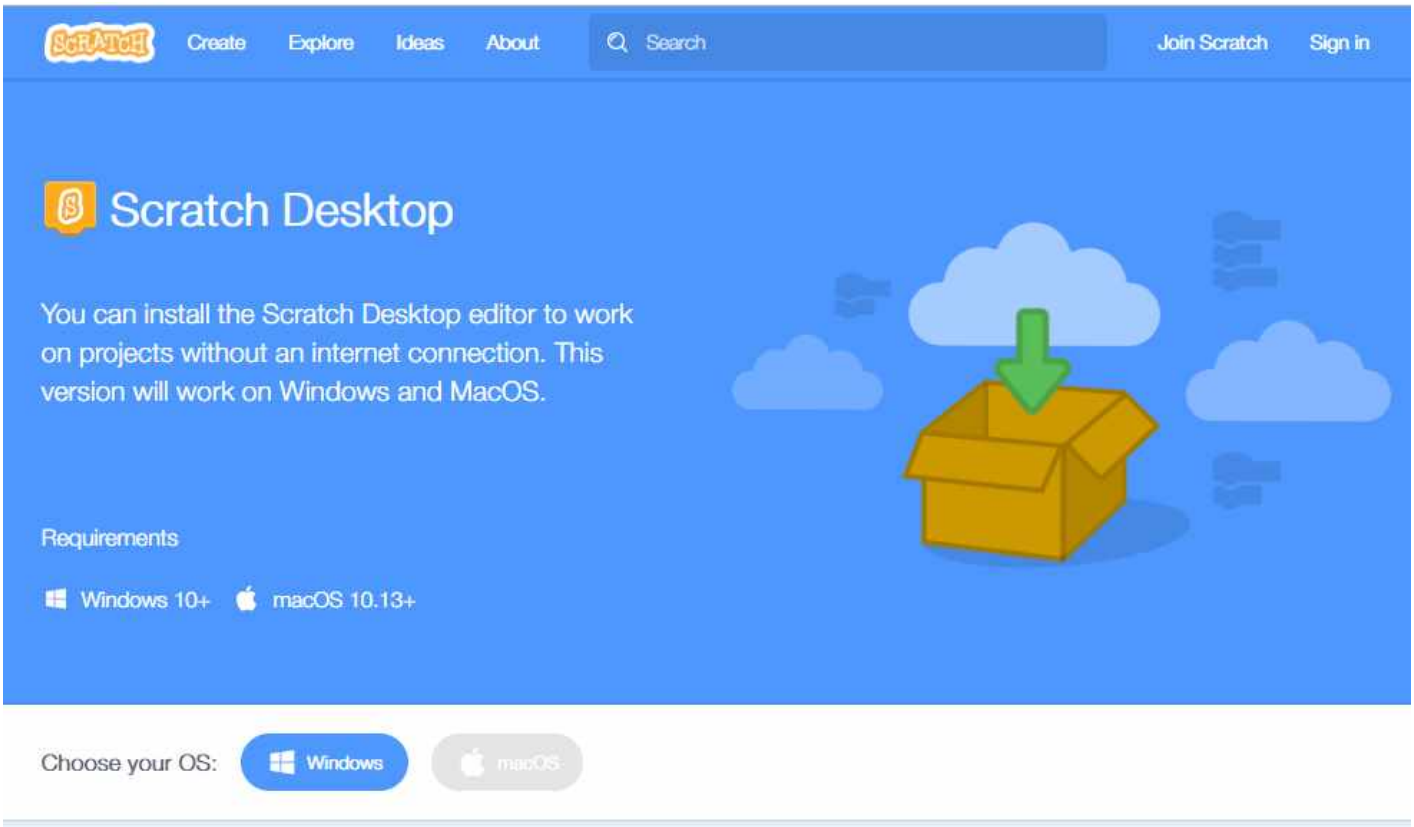
1.4 Offline Scratch-LegoWedo

La versión Offline tiene la ventaja de la rapidez en configuración y ejecución pero la desventaja de no tener los programas online, (con el riesgo de no estar disponibles, perderlos,,) pero sí que se pueden compartir.

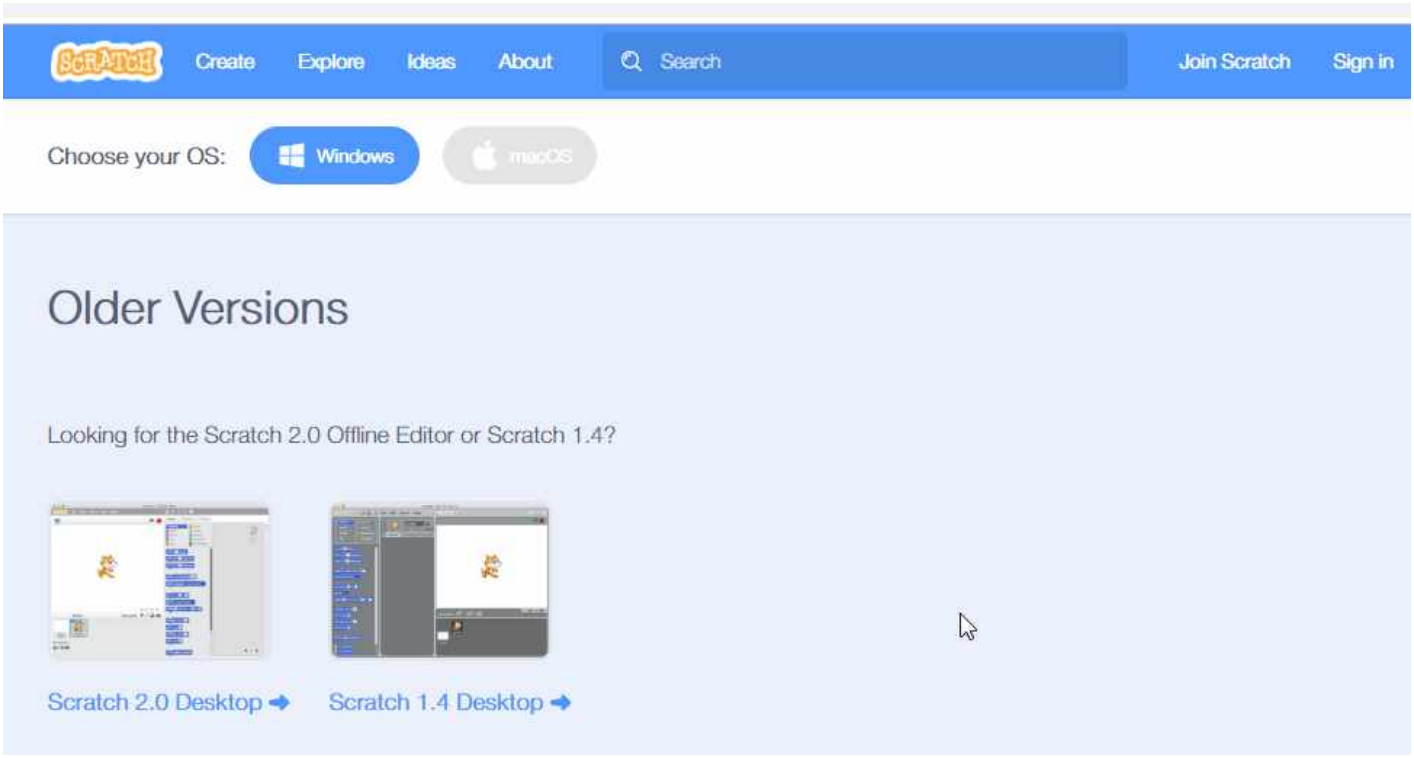
Descarga

Entramos en https://scratch.mit.edu/download y descargamos

> no existe la versión Linux :(



RECOMENDAMOS LA VERSIÓN 1.4 ir abajo



Configuración en SCRATCH 1.4

https://docs.google.com/presentation/d/e/2PACX-1vRGYuyknq9OezLYfuLoyz2t0U9MfPEYuxfJkDvXPKDqaxM50qNy2rlQX5fuQlWKC6w0kQHOqfBRHq3u/embed?start=falseloop=false&delayms=3000

Configuración en Scratch 2.4

https://docs.google.com/presentation/d/e/2PACX-1vRyGn4n_jbFdznhSnMRrv94DWQoYN9zn2eE354wSQcJciA-yEK8tIhOHpb3CY48Izmrl-6jeYjaCtZu/embed?start=falseloop=false&delayms=3000

1. ¿Qué es Lego Wedo?

1.5 LegoWedo parts

Las piezas más fundamentales que tiene este kit son:

Motor Permite que las cosas se muevan evidentemente ;)

Sensor de distancia.detecta la distancia del objeto que está frente a él. Da valores desde 0 a 100, el máximo alrededor de 30 cm). El origen 0 está algo alejado

---

Sensor de inclinación Devuelve estos valores según la inclinación:

- 0 = Sin inclinación, horizontal - 1 = Inclinado hacia abajo - 2 = inclinado hacia la derecha
- 3 = inclinado hacia arriba - 4 = inclinado hacia la izquierda

Interface:Conexión de los anteriores componentes al ordenador

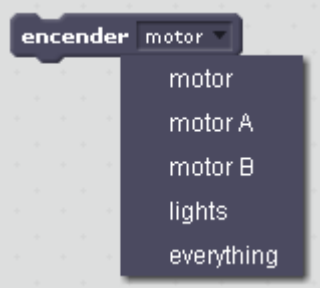
Fuentes https://wiki.scratch.mit.edu/wiki/LEGO%C2%AE_WeDo%E2%84%A2_Construction_Set

Notas:

- Se pueden adquirir aparte si se rompen o se pierden
- **No** se puede poner **los tres a la vez** (motor+sensor distancia+sensor inclinación)
- El sensor de inclinación **sólo proporciona un valor a la vez** (es decir, si está a la vez hacia arriba y a la derecha sólo dará o 2 o 3), en el interior es un objeto que según la gravedad abre o cierra el contacto de 4 interruptores (que mandan los códigos 1,2,3,4) y no puede cerrar dos a la vez.

1.6 Instrucciones LegoWedo parts

Encender motor

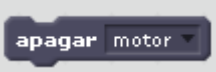


- Enciende el motor
- Tenemos varias opciones, motor A o B por si tenemos dos motores conectados (A o B según donde estén conectado en la interface, ver última fila)
- Si tuvieramos LegoWedo, también tiene luces pero en este curso lo omitimos

Fijar fuerza del motor

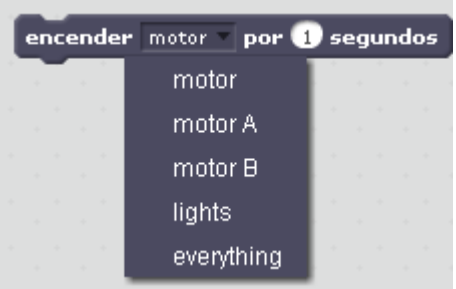
- Prepara el motor a una fuerza (equivalente a la velocidad) determinada
- Valor mínimo 1 y máximo 100
- Esta instrucción se tiene que utilizar después de la anterior.
- Ejemplo:

Apagar el motor



- Apaga el motor.

Encender motor por



Permite encender el motor por un tiempo predeterminado
Esta instrucción es equivalente a todos los sentidos a estas tres:

Fijar dirección del motor.

Fija el sentido de giro del motor (la traducción del español no la vemos muy acertada, pero nosotros te la traducimos ;)

- hacia acá = sentido de las agujas del reloj
- hacia allá = sentido contrario a las agujas del reloj

- reversa = sentido contrario al indicado en la última instrucción

Cuando inclinación sea.

- Empieza a ejecutar el programa cuando el sensor de inclinación devuelve un 1 (es decir, cuando inclinamos el sensor hacia abajo)
- El 1 se puede cambiar por los cuatro valores del sensor 0-1-2-3-4
- Se puede cambiar el = por not=

Por ejemplo si ponemos :
not= 0 entonces ejecuta el programa cuando el sensor no está en la posición horizontal.

Cuando la distancia sea.

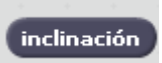
El programa correspondiente se ejecuta cuando el sensor detecta
un objeto a menos de 20

Se puede cambiar el 20 por supuesto, y el símbolo < por >

Distancia

Variable que devuelve el sensor distancia entre 0 y 100 (el 0 está algo alejado del sensor)

Inclinación



Variable que devuelve el sensor inclinación

- 0 = Sin inclinación, horizontal
- 1 = Inclinado hacia abajo
- 2 = inclinado hacia la derecha
- 3 = inclinado hacia arriba
- 4 = inclinado hacia la izquierda

2. Construcciones

2. Construcciones

2.0 Construcciones

En este módulo se tratarán las construcciones típicas de LEGO WEDO que recomienda la marca

Agradecimientos a Ro-botica.com y Aprendiendo con robótica CEIP Gil Tarín de la Muela por publicar los PDFs de las construcciones, y a mis hijos por ayudarme a construirlos

2. Construcciones

2.1 Pájaros

Objetivo

Esta construcción es simple, podría ser la primera que se construye por lo que se recomienda realizar una programación con dificultad gradual:

1. Realizarla sin interacción, sólo moviendo el motor
2. Después moviendo el motor durante un segundo
3. Cambiar el sentido de giro, primero hacia un sentido y luego que cambie
4. Interaccionar con algún objeto de Scracth
5. Interaccionar con sensor de distancia

Construcción

Lo podemos descargar aquí en pdf o en este enlace Dropbox

Propuesta

Que los pájaros bailen en un sentido y luego en otro

https://www.youtube.com/embed/OeQmYrrp61E

Solución

Otras propuestas

Se acerca la mano y los pájaros dan vueltas

https://www.youtube.com/embed/6AWfzHgvX3I

Se acerca la mano y hacen giro hacia un sentido y luego otro

https://www.youtube.com/embed/gxkK-kD4_9k

Interacción con algún objeto de Scratch

https://www.youtube.com/embed/b5oM7UWnNHM

2. Construcciones

2.2 Cocodrilo hambriento

Objetivo

Es sin duda la construcción que identifica este set de Lego. Se trata de un cocodrilo que tiene que abrir o cerrar la boca interactuando con el objeto que se encuentre en la boca. Aquí se trabaja:

- La entrada de un sensor
- La interacción que produce de forma real cerrando la boca
- Otras interacciones podrían ser de forma virtual: Personaje cocodrilo abriendo y cerrando la boca con sonido incorporado

Construcción

En este enlace tenemos el pdf o en este otro en Dropbox

Propuesta

De momento el enunciado que proponemos será que cierre la boca si se introduce un objeto y al cabo de 1 segundo que la abra para poder liberar el objeto:

https://www.youtube.com/embed/VX8i0Pg9DxE

Solución

Otra propuesta

Que no pare de masticar mientres esté el objeto

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

Solución

Fuente http://www.simonhaughton.co.uk/2010/06/lego-wedo-and-scratch.html

Otra propuesta

Que el caiman cierre y abra la boca varias veces cuando detecte un objeto

Solución

Fuente Codigo 21

Programa compartido por código 21 en este enlace

2. Construcciones

2.3 Chutador a gol

Objetivo

Se busca una interacción fácil entre el sensor distancia y el motor.

Construcción

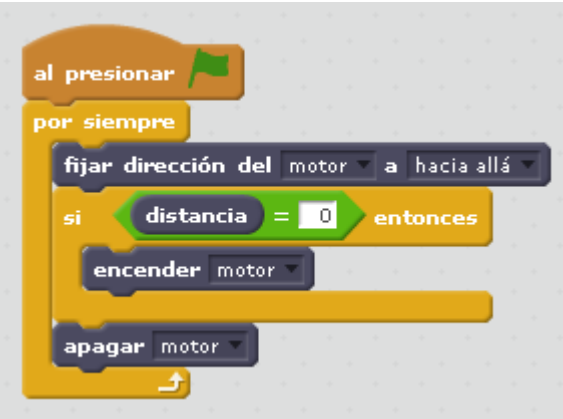
Aquí en formato PDF

Propuesta

La propuesta es fácil de adivinar ¿no? :

https://www.youtube.com/embed/jE4jhXm3YF0

Solución



2. Construcciones

2.4 Gigante colgante

Objetivo

En esta construcción se trabajan dos aspectos:

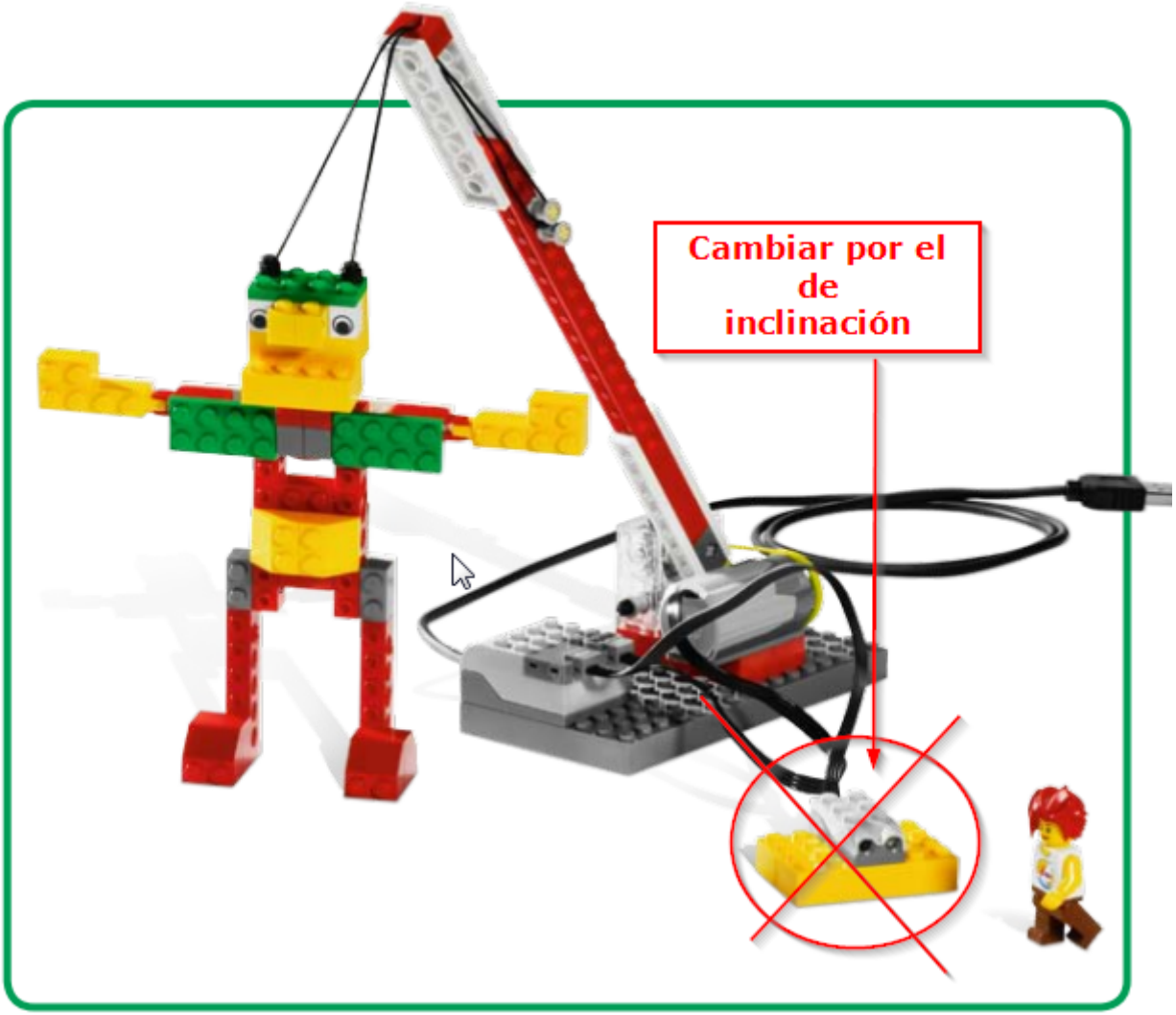
- Los mecanismos de transformación del movimiento giratorio en un multiplicador de fuerza como es el tornillo sinfin
- La programación interactuando con el sensor de inclinación

Construcción

Aquí en formato PDF o aquí en Dropbox, pero CON UNA OBSERVACIÓN aconsejamos utilizar el sensor **inclinación** en lugar del de distancia ¿por qué? porque hemos experimentado que el control de la grúa con el sensor distancia es muy difícil, no sólo en programación, sino también en sensibilidad, provocando que la grúa se pase de los límites de construcción (por ejemplo que baje más bajo que la horizontal) y provoca que las piezas sufran tensión saltando la construcción.

Aunque a estos les ha ido bien con el de proximidad.

Otra cosa: Si no se construye bien, el motor va frenado y no va.



Propuesta

Nuestra propuesta como hemos advertido antes es utilizando el sensor inclinación: Según su valor inclinado hacia arriba o hacia abajo, así actua la grua levantando al gigante:

https://www.youtube.com/embed/0JXhyMYaL_g

Solución

La puedes descargar aquí (sb2 - 54.07 KB).

Otra propuesta

Levantarlo con el teclado:

https://www.youtube.com/embed/oKlWSBhQL5M

2. Construcciones

2.5 Animadores

Objetivo

Interacción con el sensor distancia y conversión de movimiento rotatorio con oscilante

Construcción

En formato PDF o en Dropbox

Propuesta

Si ponemos un obstáculo delante del sensor, los animadores empiezan a bailar:

https://www.youtube.com/embed/bUR5CcVPD4A

Solución

Se ha añadido un poco de música para animar más:

2. Construcciones

2.6 León

Objetivo

Trabajar el sensor de inclinación con la interacción de una construcción

Construcción

En formato PDF y en Dropbox

Propuesta

Realizar un programa que al inclinar el sensor de inclinación, el león se levante y ruga. Al inclinarlo hacia abajo, el león se tumbe:

https://www.youtube.com/embed/aQSpXdlQeG4

Solución

La solución te lo puedes descargar aquí (sb2 - 61.34 KB). y el código es el siguiente:

se ha reducido la velocidad pues iba muy rápido y desmontaba la construcción

y sí... ya sé que no es un rugido, es un ladrido, pero no hay otra cosa en la biblioteca, y mi voz no es muy grave

Propuesta

La idea es de esta página web, propone poner el sensor distancia en el hocico

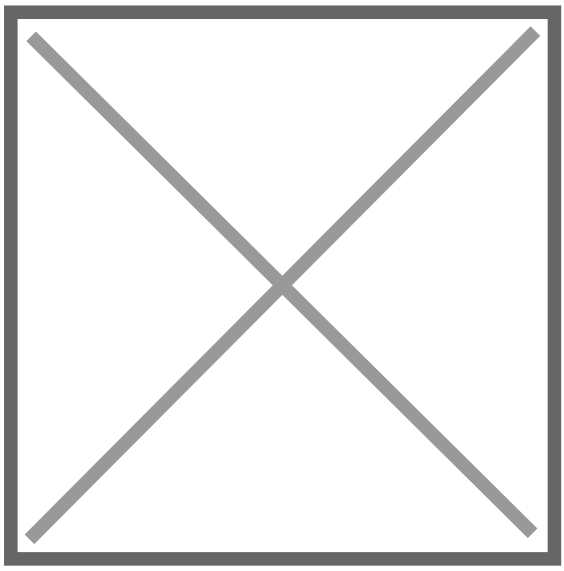
De esta manera ya le podemos dar más interacción:

https://www.youtube.com/embed/_IbYoL0eTvw

Solución

Una solución es esta:

O esta:



O aquí en Scratch: https://scratch.mit.edu/projects/38093952/

https://scratch.mit.edu/projects/watch?v=38093952/?autostart=false

2. Construcciones

2.7 Peonza

Objetivos

La peonza inteligente es un buen ejemplo de cómo la tecnología puede facilitar la ejecución de algo manual

- Utilización del sensor distancia
- Cálculo de la distancia de forma experimental para su ejecución
- Motor y engranaje multiplicador para el aumento de la velocidad de rotación

Construcción

En formato PDF en este enlace y también en Dropbox

Propuesta

La peonza tiene que girar cuando la distancia sea tal que detecte que esta encima de una superficie

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

Solución

El valor de 5 es arbritario, es la distancia tal que detecte que hay una superficie para rodar la peonza

Otras propuestas

Es que interactue con elementos del Scratch como vemos en el vídeo

https://www.youtube.com/embed/pZGX6fgNY7k

2. Construcciones

2.8 Mono percusionista

Objetivos

Aquí se busca la interacción con los objetos, y es recomendable interactuar con Scratch en los disfraces y sonidos

- Interacción sensor distancia
- Movimientos no rotativos, mecanismo de un movimiento oscilante
- Interacción con objeto de Scratch en disfraz y en sonido

Construcción

Aquí en formato PDF o en Dropbox

Propuesta

El mono toca el tambor si se le acerca el objeto de percusión, y además en el scratch interactua con el objeto mono cambiando disfraz y con sonido de percusión

https://www.youtube.com/embed/95UgMZjiN1g

En el caso anterior se ha elegido el mono que sale en la biblioteca predeterminado, pero también existe la opción de dibujar uno:

https://www.youtube.com/embed/SRSKdzvO3cI

Solución

El programa te lo puedes descargar aquí con el objeto gorila y el sonido

2. Construcciones

2.9 Pájaro

Objetivo

El principal objetivo de esta actividad es interactuar con el sensor de inclinación, su respuesta tiene que ser en algún objeto de Scratch pues esta construcción carece de motor.

Construcción

Aquí lo tienes en formato PDF o en Dropbox

Propuesta

Nuestra propuesta es una interacción con sonido y con disfraz del objeto en Scratch:

https://www.youtube.com/embed/Yb4DpYhfSaY

Solución

Aquí el programa para descargar en formato Scratch pajaro.sb2

2. Construcciones

2.10 Portero

Objetivo

Esta construcción trabaja muy bien la gamificación: la temática del futbol es una buena motivación y las posibilidades son varias fomentando la creatividad.

- Transmisión: Se transforma el movimiento rotatorio del motor en un movimiento alternante del portero
- Sensor distancia como detector de eventos: El gol
- Interacción con Scratch

Construcción

Aquí en formato PDF o en Dropbox

Propuesta

- Si marcas un gol, el personaje de Scratch cambia de disfraz (la chica cambia de las manos bajadas a las manos subidas)
- También si marcas un gol, la chica dice **gol** y suena una música Podemos introducir programación un poco más avanzada, con envío de mensajes de objetos a otros:

  - Si marca gol, se muestra una pelota durante unos segundos

  https://www.youtube.com/embed/tRoNmoAuG2I

  Ampliación: Poner un marcador (variable que contabilice los goles)

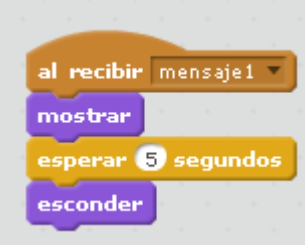
  ### Solución

  Aquí tienes el programa descargado (sb2 - 145643 B). en formato sb2

Son dos objetos, la chica (que tiene dos disfraces) y la pelota:

La chica tiene el programa principal, y envía un mensaje:

El mensaje es recogido por el objeto pelota que tiene este programa:



Otra propuesta Otra propuesta es que el portero se mueva según las fechas izquierda y derecha, de esta manera juegan dos jugadores

https://www.youtube.com/embed/7z6DfSoyKLA

Solución

Aquí en vez de mover contínuamente el portero, es según las flechas:

El programa te lo puedes descargar aquí (sb2 - 145712 B).

Reflexión ## Y si tuvieramos dos Lego Wedo Uno el portero, y otro dispara !

https://www.youtube.com/embed/UVWIL5hi33U

2. Construcciones

2.11 Avión

Objetivos

Se trabaja la interacción con Scratch, especialmente el control del movimiento y el sensor de posición

Construcción

Aquí en formato PDF o en Dropbox

Propuesta

Realizar el movimiento de un avión según el valor del sensor de desplazamiento, con movimiento de la hélice del avión y sonido (ya que estamos!)

https://www.youtube.com/embed/VSClG9h6L9c

Solución

Aquí te puedes descargar el programa (sb2 - 17650 B).

2. Construcciones

2.12 Barco

Objetivo

- Aprender los mecanismos de transformación de movimiento rotatorio del motor al ondulatorio del barco
- Programación interactiva del sensor de rotación con un objeto de Scratch

Construcción

Aquí en formato PDF o aquí en Dropbox

Propuesta

Realizar la construcción del barco navegante, y también otro en el Scratch, los dos se deben de mover sincronizados gracias al sensor de inclinación colocado en el casco del barco

https://www.youtube.com/embed/oHD47bO8t48

Solución

Son dos programas, uno que mueve el motor y otro que gira el objeto del Scratch según el sensor inclinación

Se podría hacer todo en un bucle

Te lo puedes descargar aquí (sb2 - 274.12 KB).

3. Creando

3. Creando

3.1 La creatividad

"... el aburrimiento es la antesala de la creatividad"

En este módulo te vamos a invitar a saltarnos las construcciones típicas

 Photo by Amaury Salas

3. Creando

3.2 Sensor inclinación

Lo sencillo a veces es más didáctico

Vamos a ver unos ejemplos de que no es necesario hacer construcciones para realizar unos buenos retos de programación. LEGO WEDO tiene unos estupendos sensores de inclinación y de distancia que pueden darnos mucho juego con Scratch, incluso sin crear objetos nuevos, utilizando los de la bibliteca predeterminada.

En estos ejemplos, el objetivo no es la creatividad de la construcción, sino el **pensamiento computacional**, la programación, la lógica matemática !!

Propuesta

Que se mueva la mariposa en función del sensor de inclinación de derecha a izquierda tal y como se muestra en el vídeo. La mariposa es de la biblioteca predeterminada de objetos de Scratch:

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

Solución

- En este enlace pero algunos componentes no corresponden al equivalente castellano y hay que cambiarlos.

- En este archivo (sb2 - 49.74 KB)., corregido lo anterior.

https://scratch.mit.edu/projects/watch?v=1789934/?autostart=false

El sensor (es una chorrada, pero queda guay) tiene este código

Y el sensor tiene los apropiados disfraces:

3. Creando

3.3 Más del sensor inclinación

Continuamos

Se le puede sacar mucho juego a este sencillo sensor

Propuesta

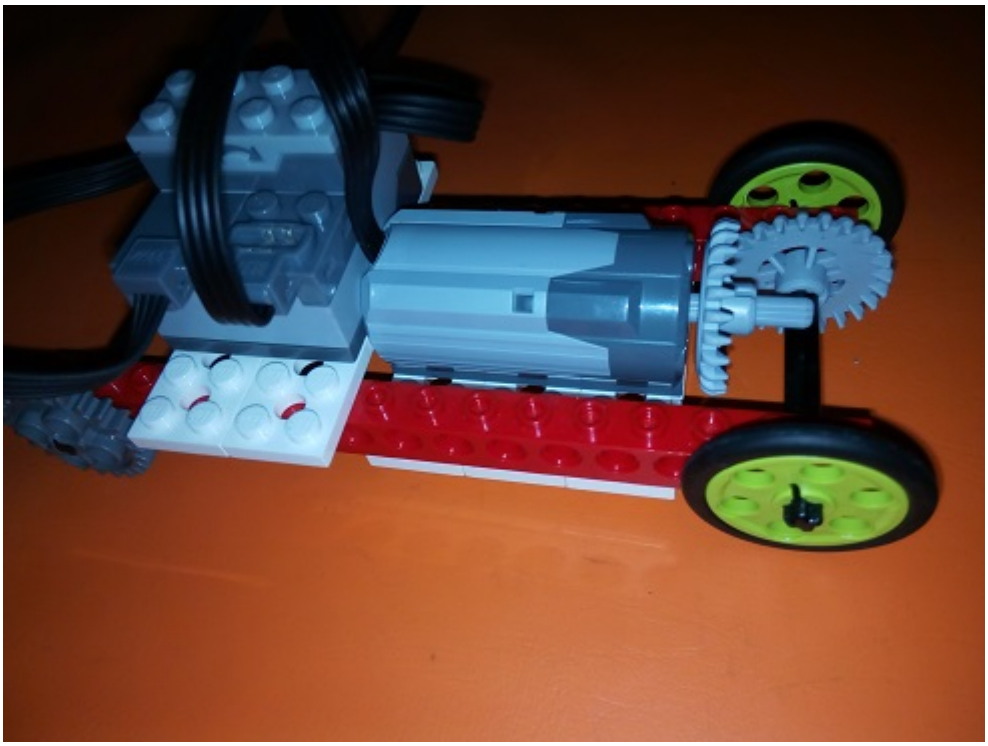
Desafiando la gravedad

Podemos hacer un coche, con el sensor de inclinación, y según su valor, actúe en contra de la inclinación, es decir que si se le inclina hacia bajo, el coche quiere subir, y al revés:

https://www.youtube.com/embed/W_NCek-rD28

Idea de Labdocente

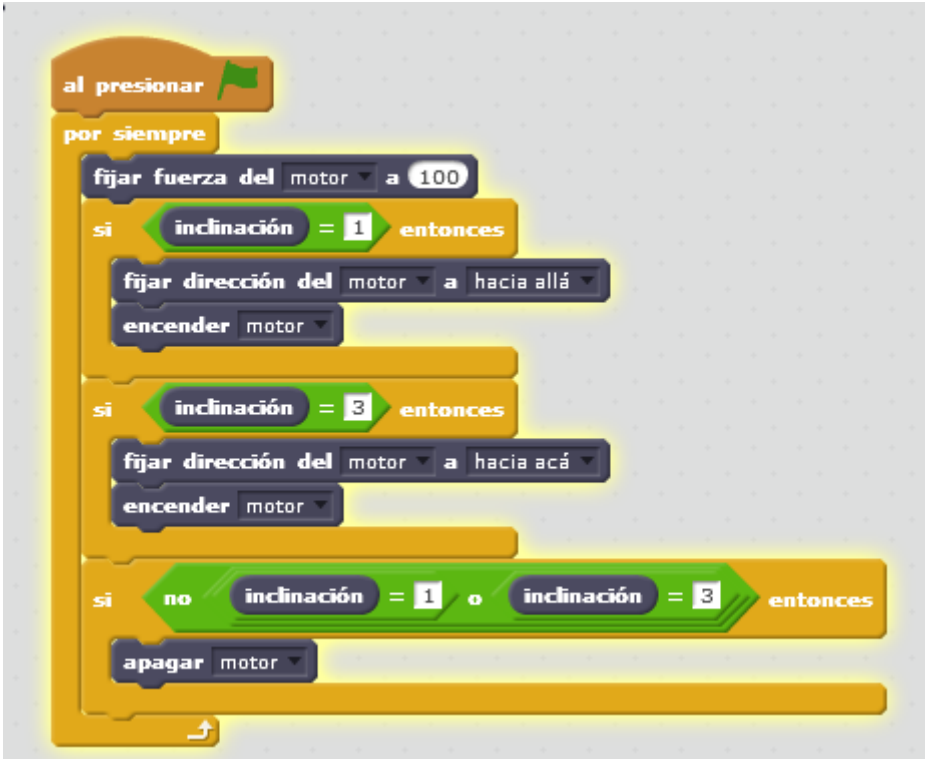
Bueno, el coche es diseño casero, sin complicaciones, pues aquí el objetivo es la sencillez en construcción para enfocar nuestra atención en la programación:



Solución

La solución tiene que actuar el motor en tres casos:

- Cuando está inclinado hacia abajo, tiene que moverse hacia un sentido
- Cuando está inclinado hacia arriba tiene que moverse hacia el otro sentido
- Cuando no está inclinado hacia arriba o hacia abajo tiene que estar quieto



Aquí si te lo quieres descargar (sb2 - 54.02 KB).

Propuesta

El siguiente programa es muy avanzado, pero nos da idea de las posibilidades del sensor de inclinación Lego WeDo

https://www.youtube.com/embed/q6d9vtaSGgA

Solución

En el siguiente enlace o bien en este archivo (sb2 - 82651 B).

https://scratch.mit.edu/projects/watch?v=583760/?autostart=false

El programa principal lo tiene la pelota que interactua con el laberinto de color azul, luego según el sensor de inclinación y según toque el color azul, cambia los valores de velocidad:

3. Creando

3.4 Sensor distancia

Propuesta

La propuesta es realizar un personaje en Scratch que sea una mariposa, con dos disfraces: Alas arriba y alas abajo (en la biblioteca tienes uno predeterminado : **Butterfly1**

Esta mariposa tiene que cambiar de disfraz según el valor del sensor de distancia (que podemos fijarlo de 0-50 alas arriba y de 50-100 alas abajo por ejemplo)

https://www.youtube.com/embed/6oyyXk2ijxM

Solución

- En este enlace aunque algunos bloques hay que sustituirlos por el equivalente en castellano

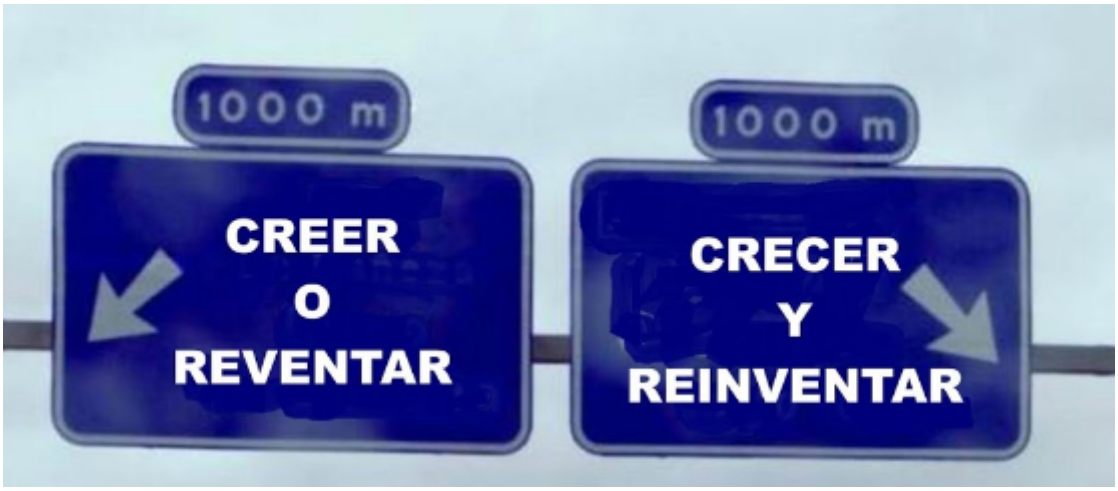
- En este archivo (sb2 - 38.27 KB). (corregido lo anterior)

https://scratch.mit.edu/projects/watch?v=1789931/?autostart=false

Fuera de la propuesta, el ejemplo tenía una barra azul que visualiza la posición del sensor a distancia, su código es el siguiente:

3. Creando

3.5 Reinventar



Es una buena ocasión para explicar los conceptos de **compartir código, software libre,...** y como todos nos beneficiamos de los pasos creativos de todos gracias a compartir libremente en beneficio de todos.

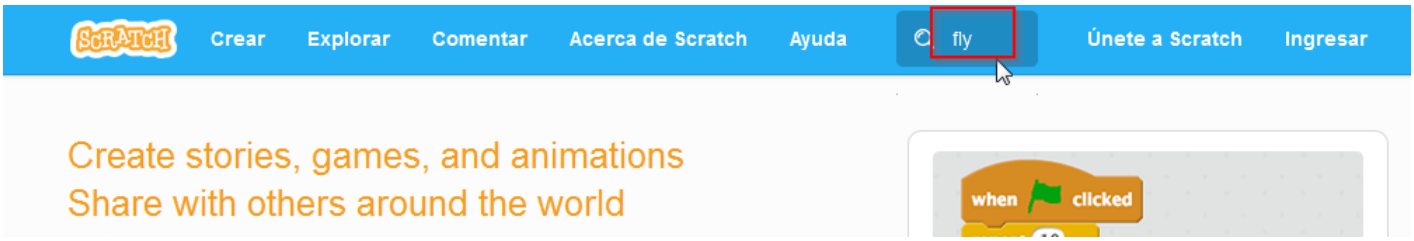
Propuesta

Objetivo

Reutilizar código ya creado, interpretarlo y reinventarlo utilizando nuestro sensor distancia LEGO WEDO

Búsqueda

Vamos a buscar algún juego sencillo, por ejemplo el típico de vuelo esquivando objetos, para ello buscamos en Scratch con la palabra clave **fly (siempre se tienen mejores búsquedas en inglés que en español, aquí podemos educar en el concepto de la globalización** y la importancia del conocimiento de las lenguas extranjeras.



elegimos alguna propuesta interesante, elegimos este:

Propuesta

Se nos ocurre que en vez que el gato suba o baje según la posición del ratón, que sea con nuestro sensor distancia de LEGO WEDO :

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

Solución

Entramos en el código y lo interpretamos, localizamos la parte responsable del movimiento del gato :

sustituimos ese código por nuestro sensor distancia

ATENCIÓN hay que hacer algo de matemáticas:

Si nuestro sensor distancia va desde 0 a 100 y el gato tiene que ir desde -150 a 150 (esto se ve moviéndolo, y abajo se visualizan las coordenadas) entonces ¿qué código hay que poner? como siempre es una recta:

y = m + n*distancia*

Si distancia=0 y tiene que ser 150 luego m=150
Si distancia =100 y tiene -150 luego n=-3

*Solución y = 150-3*distancia

(También podría ser al revés que cuando distancia=0, y=-150 y cuando distancia=100 y=150 entonces y=3*distancia-150)

La solución se puede descargar aquí (sb2 - 145.97 KB).

Se puede seguir modificando el código, por ejemplo traducir los mensajes, subir las vidas ...

3. Creando

3.6 Matemáticas, música y Lego

Propuesta musical

Realizar un programa que según la distancia toque una nota

Hay que convertir el valor que devuelve el sensor (vamos a llamarlo **distancia** de 0 a 100) en un valor nuevo (**A**, de 48 a 72).

El valor 48 corresponde a la nota DO en una escala grave y el valor 72 a la nota Do

Fuente LabDocente

https://www.youtube.com/embed/fhqpWvoDte0

No te rias! no sé tocar el piano !!! (el autor)

Solución

Aquí entran en juego las matemáticas:

La ecuación entre **A** y **distancia** corresponde a una proporción directa, es decir a una línea recta, con corte de ordenada **m** y pendiente **n** :

A = m + n*distancia*

- *Para A=48 y distancia=0 tenemos que m=48*
- *Para A=72 y distancia=100 tenemos que n = (72-48)/100= 0.24*

*Luego A= 48 + 0.24*distancia**

Descarga del programa (sb2 - 54.09 KB).

Fuente LabDocente

Fonografo

Otra alternativa es construir un toca-discos de manera casera, para que los alumnos también comprendan el funcionamiento de los reproductores de vinilo

https://www.youtube.com/embed/yj-X21rgOuw

Solución

La construcción paso a paso está en la siguiente página http://makezine.com/projects/make-35/lego-phonograph/

3. Creando

3.7 Cajón de sastre

En esta página http://www.wedobots.com/ puedes encontrar construcciones muy originales!

Por ejemplo el **León Marino**:

Instrucciones de montaje

https://www.youtube.com/embed/VJUYPehtW24

Pero aún hay más!!!

Unos cuantos ejemplos encontrados en Youtube:

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

https://www.youtube.com/embed/tvVpnfFNHF4

https://www.youtube.com/embed/tqKhD5LG6kI

https://www.youtube.com/embed/84MCVrS-DB8

https://www.youtube.com/embed/r4r8spv_VF4

https://www.youtube.com/embed/BqzA2y0A7bc

https://www.youtube.com/embed/UPBR73Uld3g

https://www.youtube.com/embed/0zYJZvPYUqo

https://www.youtube.com/embed/PZ1ejFnzv4s

https://www.youtube.com/embed/1wu45RO2yog

https://www.youtube.com/embed/DX_9SrqXKQo

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

https://www.youtube.com/embed/vZ2WoC-eM3E

https://www.youtube.com/embed/MnFe9YpT0-s

https://www.youtube.com/embed/lfnbI-TA2XQ

https://www.youtube.com/embed/WR6oFVDVx-s

3. Creando

3.8 Tu cajón de sastre

https://padlet.com/embed/krn59al7ll9l

Hecho con Padlet

Creditos

2017 por CATEDU (Javier Quintana Peiró).

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