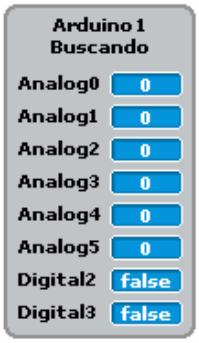
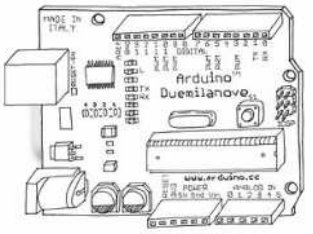
ACTIVIDADES CON S4A (SCRATCH PARA ARDUINO)



A.1.- Actividades previas

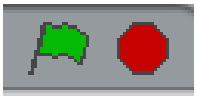
- Cargar el firmware de S4A (S4AFirmware16.ino) en la placa Arduino mediante el IDE de Arduino
- Conectar la placa Arduino al puerto USB y abrir el programa S4A
- Ocultar y mostrar la tabla de sensores de la placa Arduino (a través de clic derecho sobre el objeto en la lista de objetos y seleccionando la opción que corresponda).
- Esconder y mostrar la imagen de la placa Arduino. Para esconder haciendo clic sobre la pieza de bloques "esconder". Para mostrar haciendo clic sobre el bloque "mostrar" o en el menú contextual del objeto.



A.2.- Cambios de disfraces

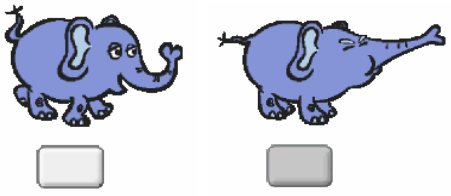
- Para las siguientes actividades mantén ocultas la placa Arduino y la tabla de sensores.
- Crea un objeto a partir de la librería de Arduino. Tendrá dos disfraces: la imagen elephant1-a y la elephant1-b (ambas están en la carpeta "Animals").Cambia el nombre del objeto y llámale "Elefante azul". Cambia el nombre de los disfraces y llámales "Andando" y "Soplando".

A.2.a.- Hacer un programa para que, a partir de que hagamos clic en la bandera verde, el elefante cambie de disfraz a intervalos de 0,5 segundos por tiempo indefinido, hasta que terminemos haciendo clic en el botón rojo. Deberás utilizar los siguientes bloques:

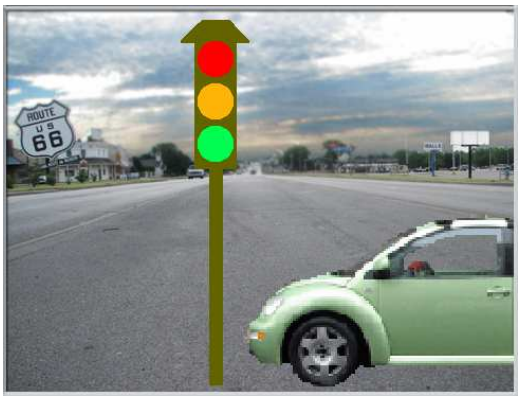


A.2.b.- Hacer un programa para que el elefante cambie de disfraz cada vez que presionemos la tecla "d" o bien hagamos clic sobre el elefante. Deberá utilizar los siguientes bloques:

A.2.c.- Crea un nuevo objeto con dos disfraces, la imagen "button" y la imagen "buttonPressed" (ambos en la carpeta "Things"). Llama al objeto "Pulsador". Haz un programa para que cada vez que haga clic sobre el objeto Pulsador cambie de disfraz y haga cambiar de disfraz también al elefante. El disfraz "button" de Pulsador debe corresponder con el disfraz "Andando" del elefante y el disfraz "buttonPresed" con el disfraz "Soplando". Inicialmente, al hacer clic sobre la bandera verde, el pulsador no estará presionado y el elefante estará andando.

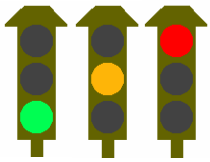


A.2.d.- Simulación de un semáforo con S4A.



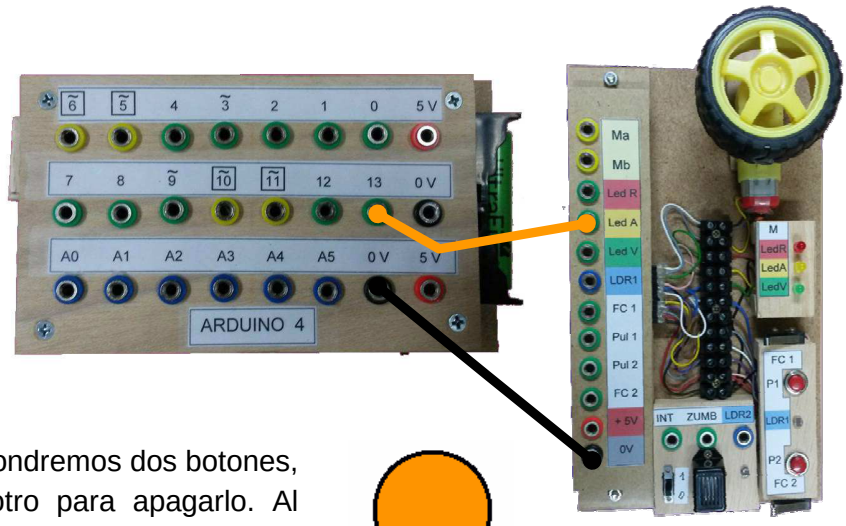
La luz verde debe estar encendida 5 s, la ámbar, a continuación, 2 s y la roja 4 s. Y así sucesivamente.

Puedes resolver este ejercicio de varias formas. La más fácil es crear un objeto con tres disfraces, como los que se muestran al margen, que son las tres posibles situaciones en las que puede estar el semáforo. El programa hará que el objeto vaya pasando de un disfraz a otro en los tiempos correspondientes.

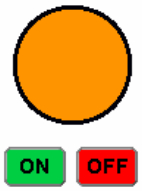


A.3.- Encendiendo y apagando un LED real haciendo clic en botones en la pantalla del ordenador.

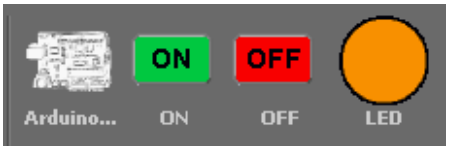
Vamos a encender y apagar un LED desde el ordenador conectado a la patilla 13 de Arduino. Para ello utilizaremos unos entrenadores que tenemos para practicar con Arduino. El montaje sería como el de la figura:



En la pantalla del ordenador dispondremos dos botones, uno para encender el LED y otro para apagarlo. Al mismo tiempo, el LED se simulará en la pantalla de ordenador con un círculo con dos disfraces, uno oscuro cuando el LED real esté apagado y otro del color del LED, cuando el LED real esté encendido.



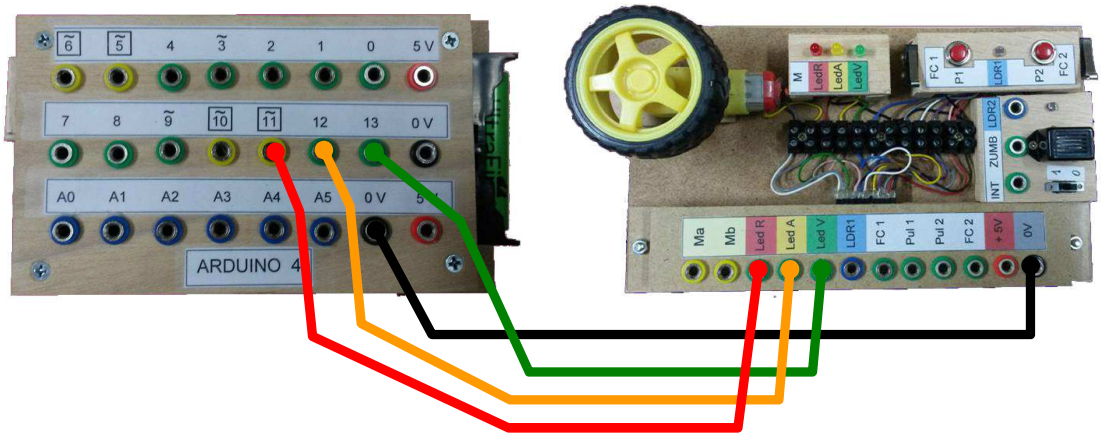
El programa tiene 4 objetos, incluido la placa Arduino.



Cada objeto tiene su propio programa. Necesitarás los siguientes bloques:

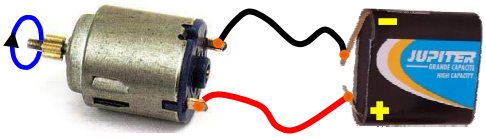
A.4.- Controlando un semáforo real con S4A.

Continúa el ejercicio A.2.d, para que, además del semáforo simulado en la pantalla del ordenador, se controle un semáforo real formado por tres LEDs conectados a las patillas 11, 12 y 13 de Arduino. Utiliza los entrenadores para Arduino.

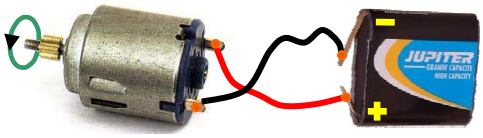


A.5.- Controlando un motor real con S4A.

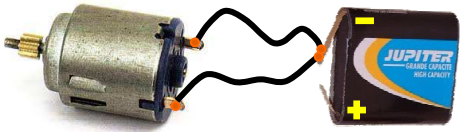
Vamos a ver ahora como podemos hacer funcionar un motor de corriente continua como los que usamos en los proyectos de Tecnología. Como sabemos, los motores pueden estar parados o pueden girar en dos sentidos, dependiendo de los polos de la pila que le conectemos a sus conectores.



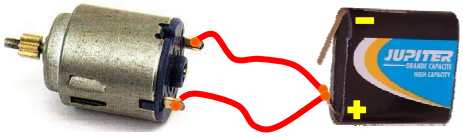
GIRA HACIA UN LADO



GIRA HACIA EL OTRO LADO

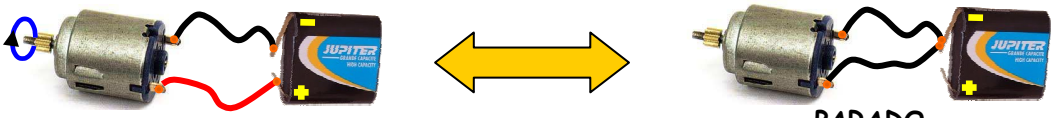


PARADO



PARADO

Si en nuestro proyecto **el motor sólo tiene que girar en un sentido**, podemos conectar uno de sus conectores fijos a tensión negativa y el otro lo conectamos a tensión positiva cuando queremos que gire y a tensión negativa cuando queremos que se pare.



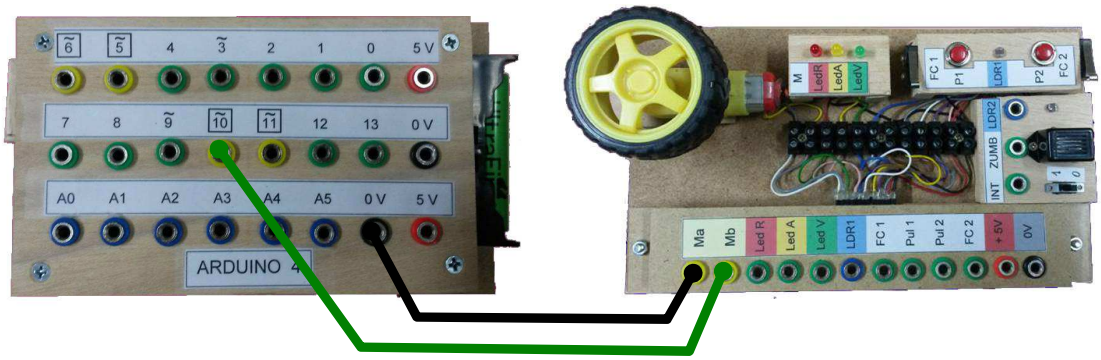
GIRA PARADO

Si el conector que cambia de polaridad del motor lo conectamos, por ejemplo, en el pin 10, y el otro conector lo conectamos directamente al borne negativo, las órdenes que tenemos que dar son:

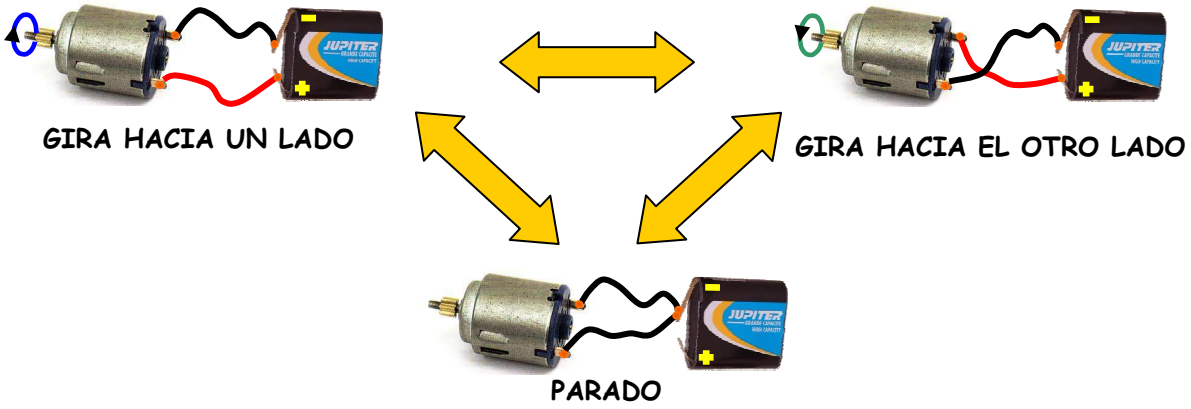
En nuestro entrenador de Arduino sólo podemos usar para conectar motores los pines 5, 6, 10 y 11.

GIRA

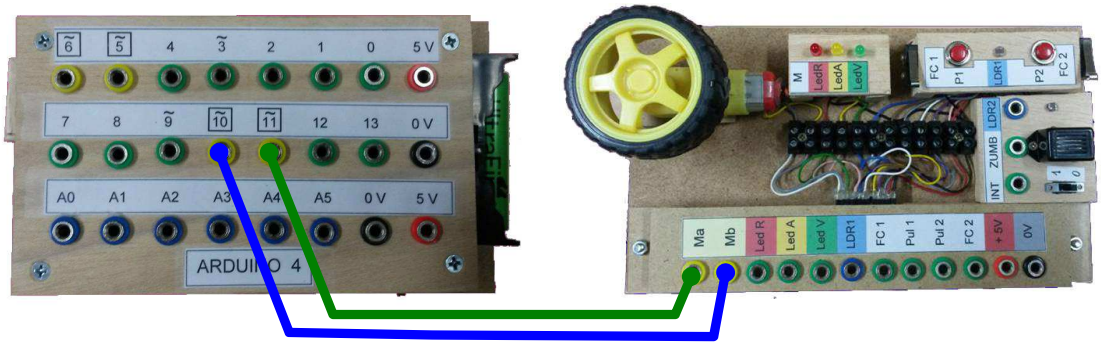
PARADO



Si en nuestro proyecto **el motor tiene que girar en ambos sentidos y pararse**, tenemos que ir cambiando el signo de la tensión aplicada a los terminales.



Cada conector del motor tiene que ir entonces en un pin de Arduino, por ejemplo, elegimos el 10 y el 11. Las órdenes serían



A.5.a.- Control de motor haciendo clic en botones en la pantalla del ordenador

Realiza un programa tal que en la pantalla del ordenador tengamos tres botones (“girar izquierda”, “girar derecha” y “parar”), de forma que cuando hagamos clic sobre uno de ellos, el motor haga lo que el botón le indica hasta que hagamos clic en otro botón.

A.6.- Control de motores con pines analógicos.

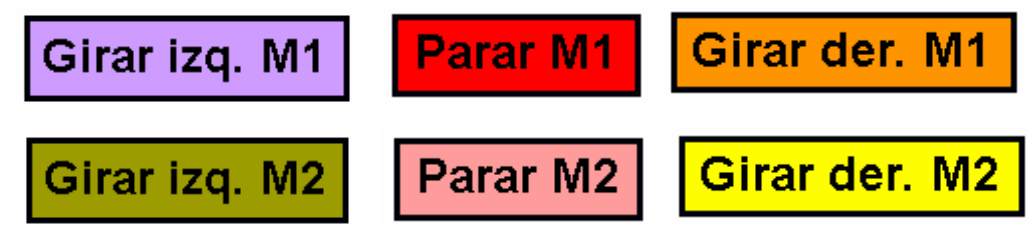
Como hemos comentado en una nota del ejercicio A.5, en nuestro entrenador de Arduino sólo podemos controlar motores con los pines 5, 6, 10 y 11, ya que son los que disponen de amplificadores suficientes para poder aportar la corriente que requieren los motores sin dañar la placa Arduino. Si ya tenemos usados los pines 10 y 11 con un motor y tenemos otro motor en el proyecto, podemos conectarlo a los pines 5 y 6 que son salidas analógicas en S4A. Lo haremos con las órdenes:

GIRA HACIA UN LADO | PARADO | GIRA HACIA EL OTRO LADO

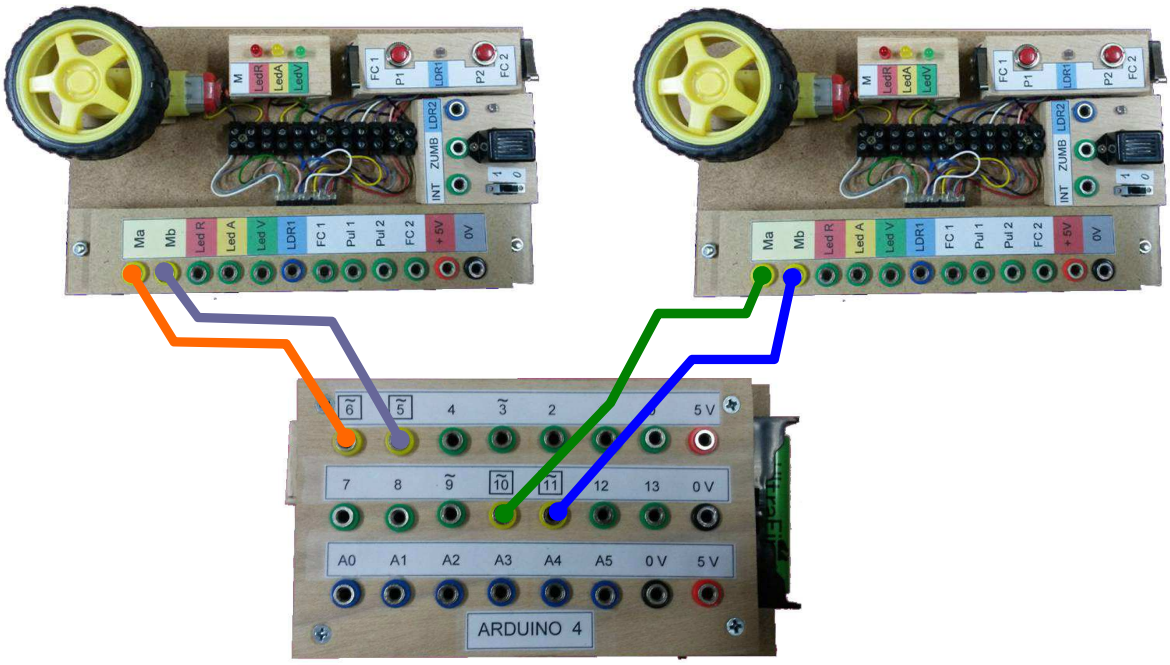
Incluso podemos conseguir que los motores vayan un poco más lentos, cambiando el número 255 por otro menor. Aunque si lo reducimos demasiado los motores por tendrán suficiente energía para moverse.

A.6.a.- Control de dos motores haciendo clic en botones en la pantalla del ordenador

Realiza un programa tal que en la pantalla del ordenador tengamos seis botones, tres para cada motor (“girar izquierda”, “girar derecha” y “parar”), de forma que cuando hagamos clic sobre uno de ellos, los motores hagan lo que el botón les indica hasta que hagamos clic en otro botón.

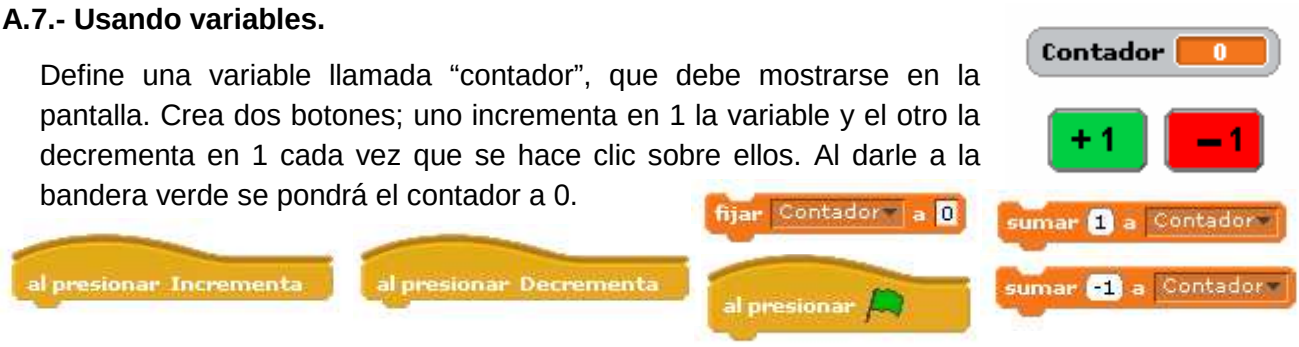


Para esta actividad tendremos que usar dos entrenadores, con las siguientes conexiones:



A.7.- Usando variables.

Define una variable llamada “contador”, que debe mostrarse en la pantalla. Crea dos botones; uno incrementa en 1 la variable y el otro la decrementa en 1 cada vez que se hace clic sobre ellos. Al darle a la bandera verde se pondrá el contador a 0.

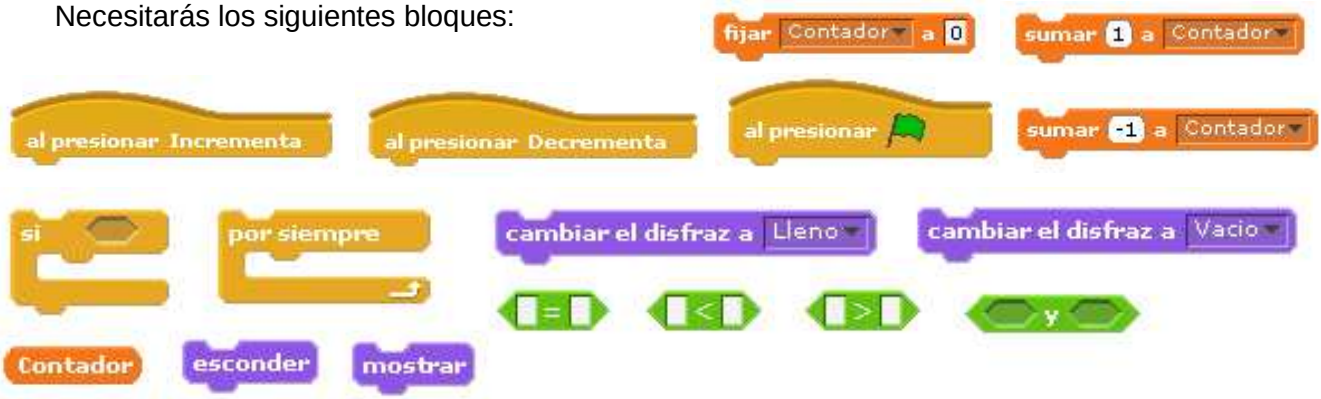


A.8.- Usando condicionales.

Supongamos que el ejercicio anterior el contador refleja el número de coches que hay dentro de un aparcamiento con una capacidad máxima de 10 plazas. El botón de incremento simula la entrada de un vehículo y el botón de decremento la salida. Cuando el contador llegue a 10 (valor máximo) ya no se incrementará más aunque se haga clic sobre el botón de incremento y aparecerá un letrero que indique “*Aparcamiento lleno*”. Cuando el contador llegue a 0 ya no se decrementará más, aunque se haga clic sobre el botón de decremento, y aparecerá un letrero que indique “*Aparcamiento vacío*”.

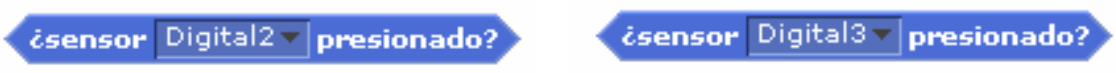
Los letreros pueden ser objetos creados directamente con S4A, que aparezcan y desaparezcan cuando corresponda. También puede ser un único objeto con dos disfraces.

Necesitarás los siguientes bloques:



A.9.- El uso de sensores digitales.

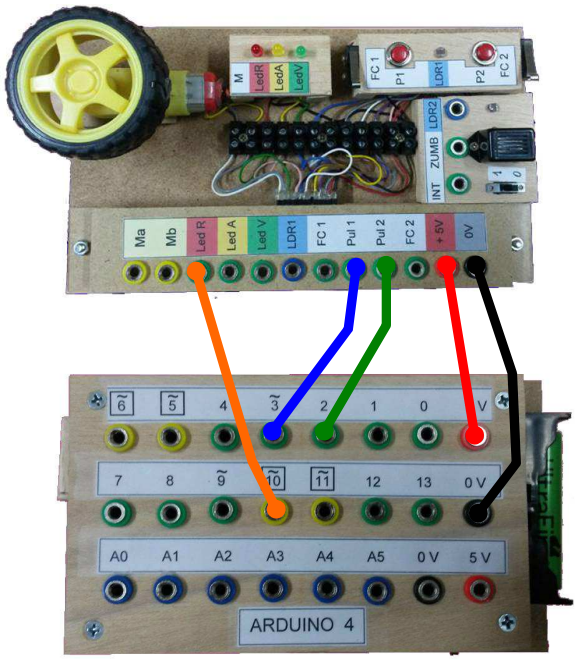
Para que nuestros programas detecten cuando están presionados los pulsadores y los finales de carrera o si los interruptores están cerrados, usamos los **sensores digitales**. S4A sólo dedica dos pines a estos sensores, el 2 y el 3.



Cuando a estos pines hacemos llegar un valor alto de tensión (5V o cercano a 5V) las funciones anteriores nos dan un valor verdadero (true) y cuando les llega un valor bajo de tensión 0V o cercano a 0V) nos dan un valor de falso (false).

Ejemplo

Queremos que un LED conectado en el pin 10 se encienda cuando se pulse el pulsador conectado al pin 2 y que se apague cuando se pulse el pulsador conectado al pin 3. El programa sería:



A.10.- Control barrera de Aparcamiento.

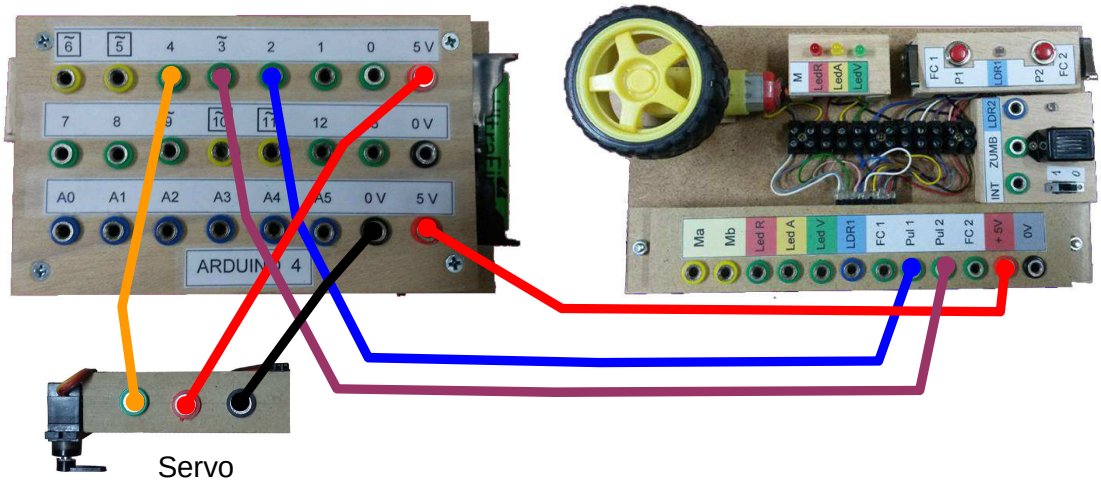
Vamos a resolver el mismo problema del aparcamiento de la actividad A.8, pero ahora el incremento y el decremento del contador se consiguen pulsando sendos pulsadores situados en el entrenador que irán conectados en las patillas 2 y 3 de Arduino.

Además, cada vez que se pulse uno de los pulsadores, se accionará un servomotor conectado al pin 4 de Arduino que mueve una barrera de parking. El servomotor girará 90º para subir la barrera, la mantendrá subida 5 segundos y girará 90º en sentido contrario para volver a bajarla.

Nota: los servomotores sólo pueden conectarse en los pines 4, 7 y 8 de Arduino cuando usamos S4A.

Además de los bloques de la actividad anterior, deberás utilizar los siguientes:

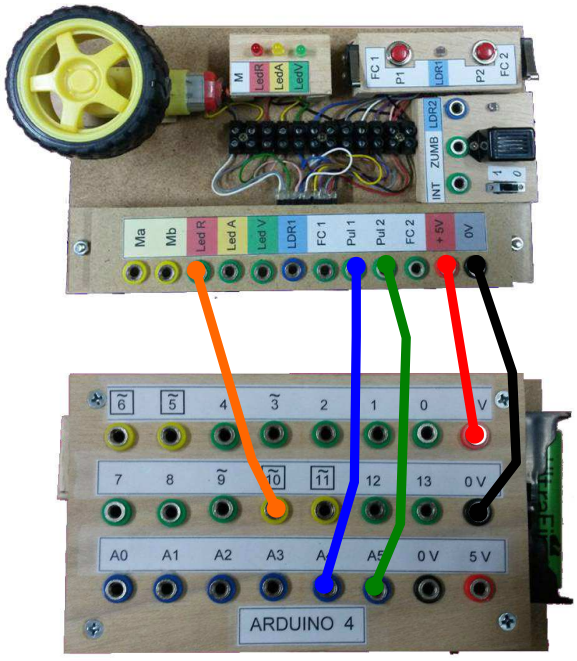
También puedes recrear virtualmente el movimiento de la barrera en la pantalla del ordenador e incluso simular la entrada o salida de un vehículo.



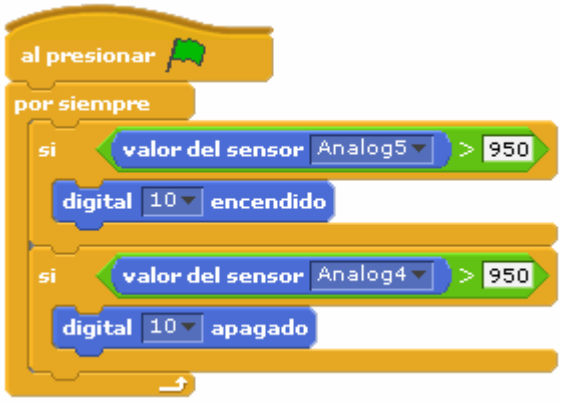
A.11.- El uso de sensores analógicos.

Estos sensores nos permiten medir magnitudes que pueden tomar un gran número de valores en un rango. Por ejemplo, el nivel de luminosidad (es medido por las LDR), la temperatura (es medido por sensores de temperatura como el LM35), etc. S4A dispone hasta 6 pines analógicos (desde A0 hasta A5). Si se le aplica 5V a estos pines, proporcionan un valor de 1023, mientras que cuando se les aplica 0V proporcionan un valor de 0. Los valores intermedios son proporcionales.

Atención: Los pines analógicos también nos pueden servir para detectar pulsaciones de pulsadores o finales de carrera. En efecto, si conecto unos de los terminales de un final de carrera a un pin analógico y el otro terminal del final de carrera a positivo, al pulsarlo le introduzco un valor de 5 V al pin con lo que si lo leo, obtendremos un valor de 1023 o muy cercano, mientras que si no está pulsado, obtendremos un valor cercano a 0.



Ejemplo: para el resolver el mismo caso de la actividad A.9, con pines analógicos, haríamos:



A.12.- Control de motor de corriente continua con LDR y finales de carrera.

Tenemos una puerta accionada por un motor. Hacer un programa de modo que al tapar una LDR se ponga en marcha el motor abriendo la puerta. Cuando se active el final de carrera de puerta abierta (FCA) se parará el motor durante 5 segundos. Tras este tiempo, el motor comenzará a girar en sentido contrario hasta activar el final de carrera de puerta cerrada (FCC).

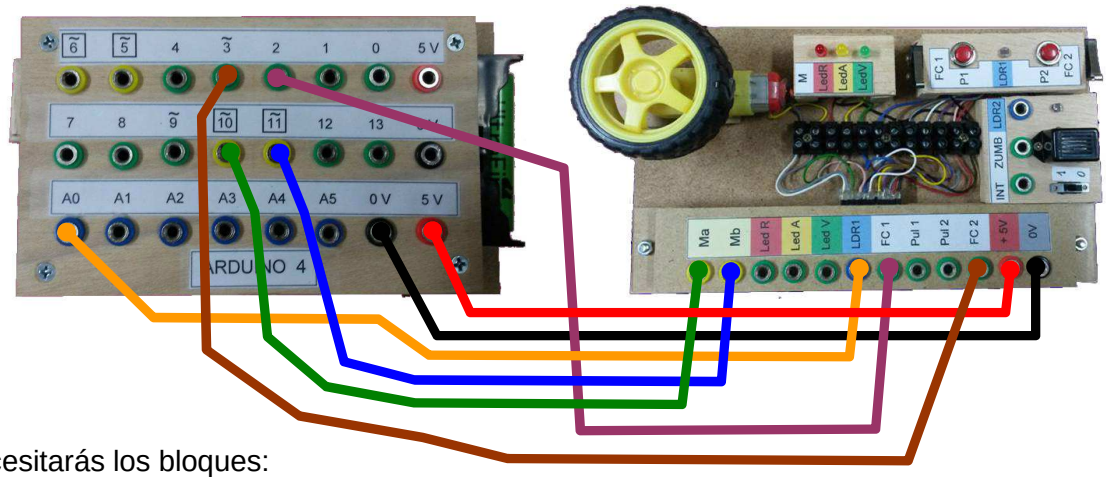
El motor irá conectado a los pines 10 y 11 de Arduino, el FCC en el pin 2, el FCA en el pin 3 y la LDR en el pin analógico A0.

Nota 1: Si haces alguna prueba con el funcionamiento de la LDR, observarás que cuanto más la tapes menor es el valor que se lee en el pin analógico donde va conectada y, por el contrario cuanto más luz le dé, mayor es el valor leído. ¿Cuándo podemos considerar que está lo suficientemente tapada de cara a nuestra actividad? Cuando el valor leído caiga por debajo de un valor límite que podemos establecer nosotros. Por ejemplo, 200.

Nota 2: Debes tener en cuenta que para controlar el giro de un motor en ambos sentidos hay que alimentar a éste desde dos pines digitales. Si ordenamos apagado por ambos pines o encendido por ambos pines, el motor estará parado. Si ordenamos encendido por un pin y apagado por el otro girará en un sentido y si cambiamos el pin encendido a apagado y viceversa, el motor girará en sentido contrario.

Nota 3: Al empezar el programa, se mirará cómo se encuentra la puerta y en caso de que no esté cerrada, se pondrá en marcha el motor hasta que quede cerrada.

Las conexiones en los entrenadores de Arduino serán las de la figura siguiente:



Necesitarás los bloques:

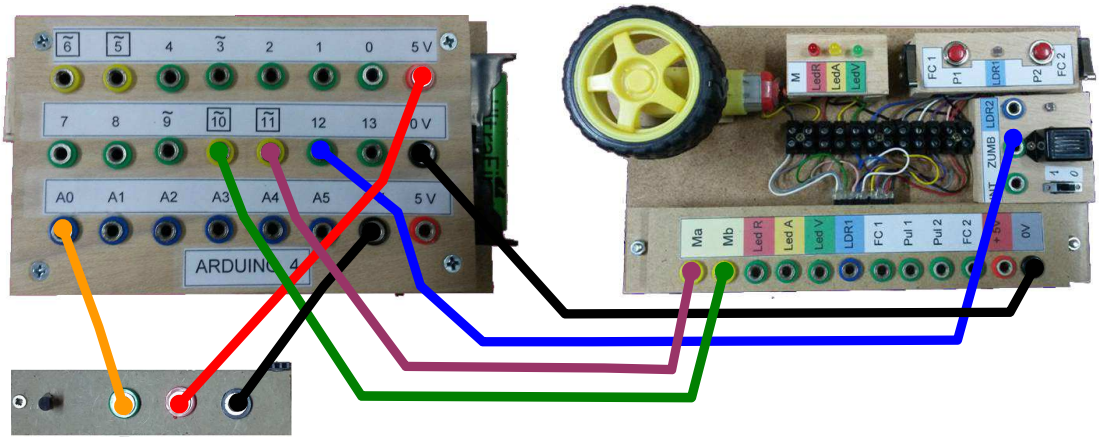
También puedes recrear virtualmente el movimiento de apertura, la espera y el cierre de la puerta en la pantalla del ordenador al mismo tiempo que tienen lugar los movimientos del motor.

A.13.- Control de temperatura con sensor LM35, accionado ventilador y alarma.

Realizar un programa de forma que mida la temperatura ambiente mediante un sensor LM35 y muestre su valor en la pantalla del ordenador. Cuando la temperatura pase de 28 ºC se pondrá en marcha un ventilador (será el motor del entrenador) y se mantendrá en marcha hasta que la temperatura baje de 28 ºC. Si la temperatura subiera por encima de 32 ºC, se pondrá a sonar un zumbador (alarma de incendio) y se parará el ventilador para no avivar el fuego.

Nota: se han elegido estas temperaturas para poder alcanzarlas fácilmente con nuestro calor corporal cogiendo con cuidado el sensor entre los dedos.

El motor irá conectado a los pines 10 y 11 de Arduino, el zumbador en el pin 12 y el sensor LM35 en el pin analógico A0.



LM35

Necesitarás los siguientes bloques (algunos más de una vez):