

Tabla de contenido

Introducción	1.1
Componentes	1.2
Piezas impresas 3D	1.2.1
Placa botonera	1.2.2
Arduino Nano	1.2.3
Motores paso a paso 28BY J-48	1.2.4
Drivers ULN2003	1.2.5
Protoboard 170 puntos	1.2.6
Buzzer 5V	1.2.7
Portapilas 4 unidades AA	1.2.8
Cables arduino macho-hembra	1.2.9
Canica 14 mm	1.2.10
Juntas tóricas 63x57x3 mm	1.2.11
Tornillería y tuercas	1.2.12
Guía de montaje	1.3
Instalar programación	1.4
Descargando IDE Arduino	1.4.1
Instalando IDE Arduino	1.4.2
Abriendo IDE Arduino	1.4.3
Descargando la configuración para Escorabot	1.4.4
Test de botonera	1.4.5
Cambiar los valores de la botonera	1.4.6
Cargar código modificado	1.4.7
Modificar programación	1.5
Modificar Velocidad	1.5.1
Modificar Distancia de avance	1.5.2
Modificar Grados de giro	1.5.3
Material para trabajar en clase	1.6
Tableros y fichas	1.6.1
Disfraces	1.6.2
Máscaras Impresión3D	1.6.3
Dados CoDices	1.6.4
Diseño de ruedas	1.6.5
Añadir módulo Bluetooth BLE	1.7
Programación con mBlock	1.8
Pensamiento computacional	1.9
Créditos	2.1

¿Cómo nace Escornabot?

[Escornabot](#) es un proyecto de código abierto que nace en Galicia.

Todo empieza en la asociación de makers [Bricolabs](#), allí coinciden tres personas:

- [Tucho Méndez](#) aporta la **idea** inicial.
- [Xoan Sampaiño](#) diseña las **piezas 3D**.
- [Rafa Couto](#) realiza la **programación para Arduino**.

Deciden crear un **robot libre** para poder trabajar en el aula la **robótica educativa** sin depender de una marca comercial.

Con el tiempo se van sumando otras personas:

- [Kabier Rosas](#) diseña las **nuevas PCB**.
- [Jorge Lobo](#) profesor y maker, aporta diversas **aplicaciones para el aula** a través de su [blog](#).
- [Miguel Gesteiro](#) desarrolla la **aplicación MUWI y Bluetooth BLE** junto a Ismael Serrano.

Entre todos van aportando una documentación que otras personas hemos ido aprovechado para difundir el proyecto.

Para comprender la filosofía del proyecto, imprescindible ver dos vídeos de **Tucho Méndez**:

El primero, en **Arduino day 2015** organizado por [Makers Lugo](#).



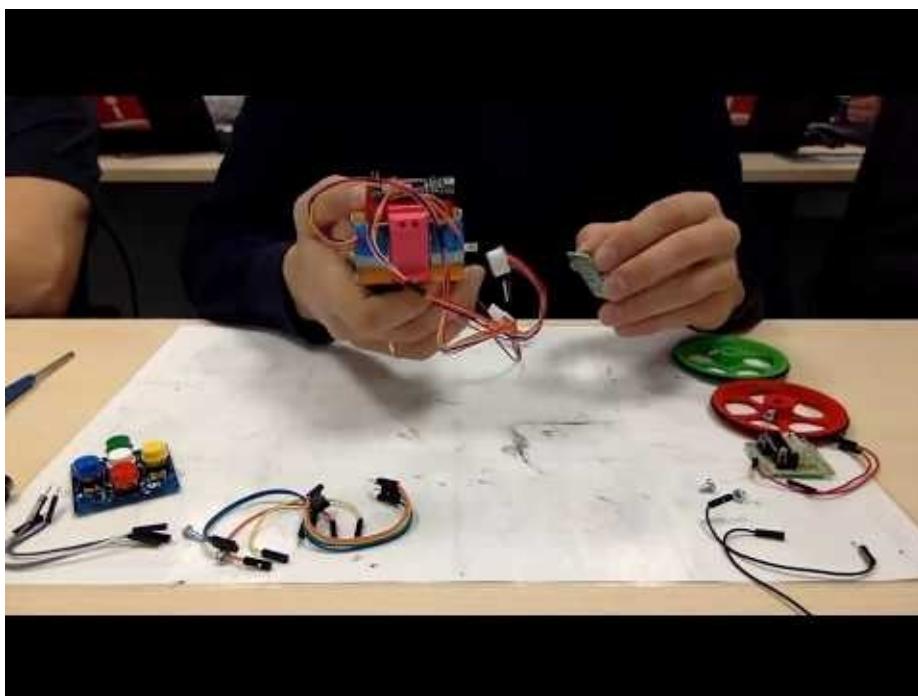
[Video link](#)

El segundo, en el **I Encuentro Coruña Dixital 2016**. Proyectos abiertos de programación y robótica educativa, la experiencia de escornabot.



[Video link](#)

En este curso haréis algo parecido a esto:



[Video link](#)

Ver [aquí](#) otra versión más cómica ;-)

Componentes Versión DIY

Puedes consultar los enlaces a todos los componentes en [esta web](#)

Hay [muchas versiones](#), nos vamos a centrar en la [versión DIY - Brivoi Audacius](#)

En este apartado veremos todos los componentes explicando de manera simple su funcionamiento, eso te permitirá solucionar cualquier problema que vaya surgiendo.

No te asustes, **no entraremos en detalles técnicos**.



- [Piezas impresas 3D](#)
- [Placa botonera](#)
- [Arduino Nano](#)
- [Motores paso a paso 28BYJ J-48](#)
- [Drivers ULN 2003](#)
- [Protoboard 170 puntos](#)
- [Buzzer 5V](#)
- [Portapilas 4 unidades AA](#)
- [Cables arduino macho-hembra](#)
- [Canica 14 mm](#)
- [Tornillería y tuercas](#)

Piezas impresas 3D

Sostienen el resto de componentes y **conforman el chasis del robot**, se pueden imprimir en diferentes materiales. El más utilizado suele ser un plástico llamado PLA por ser el más económico y simple de imprimir, con una altura de capa entre 0.2 y 0.3 mm. [AingeruJM](#) nos reporta el consejo de [Rafa Couto](#):

"Cono relleno 20%, 2 contornos, 3 capas inferiores y superiores debería quedarte bien sin esfuerzo en la mayoría de las impresoras."

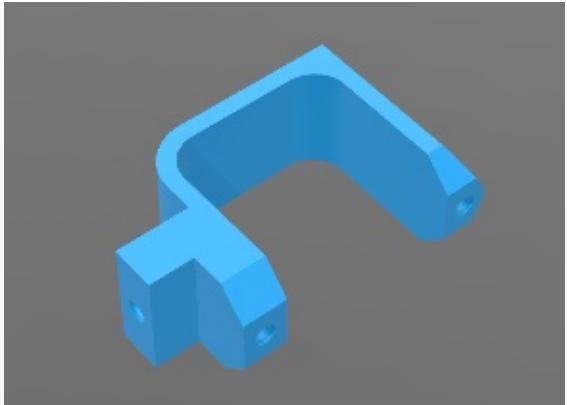
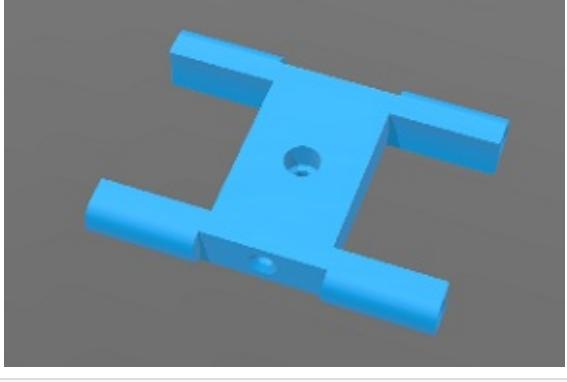
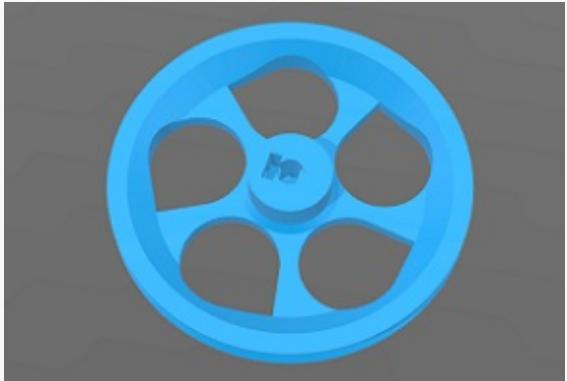
En una de la columna aparece "**nombre del archivo .stl**" es el tipo de **formato que utilizan los programas de las impresoras 3D** para poder leer e imprimir la pieza, si pinchas encima puedes descargar el archivo.

Piezas exclusivas de la versión DIY

Nombre del archivo .stl	Descripción	Imagen
PCBAddon-KeypadBracket-5Buttons.stl	Sujeción placa botonera	
board-bracket.stl	Sujeción protoboard 170 puntos	

Piezas comunes con otras versiones

Nombre del archivo .stl	Descripción	Imagen
ballcaster-v2.stl	Sujeción bola v2	

	battery-bracket.stl	Sujeción batería o portapilas AA	
	MotorBracket.stl	Sujeción motores	
	wheel-l.stl	Rueda Izquierda	
	wheel-r.stl	Rueda derecha	

Placa Botonera



Incorpora los botones de acción y **es la encargada de transmitir las órdenes de los movimientos** al cerebro (placa Arduino Nano)

A través de los tres pines acodados GND, Sig y 5V (los tres pinchos de la parte superior derecha de la imagen) se conecta a la protoboard (la veremos más abajo)

Si miramos uno a uno sus componentes:

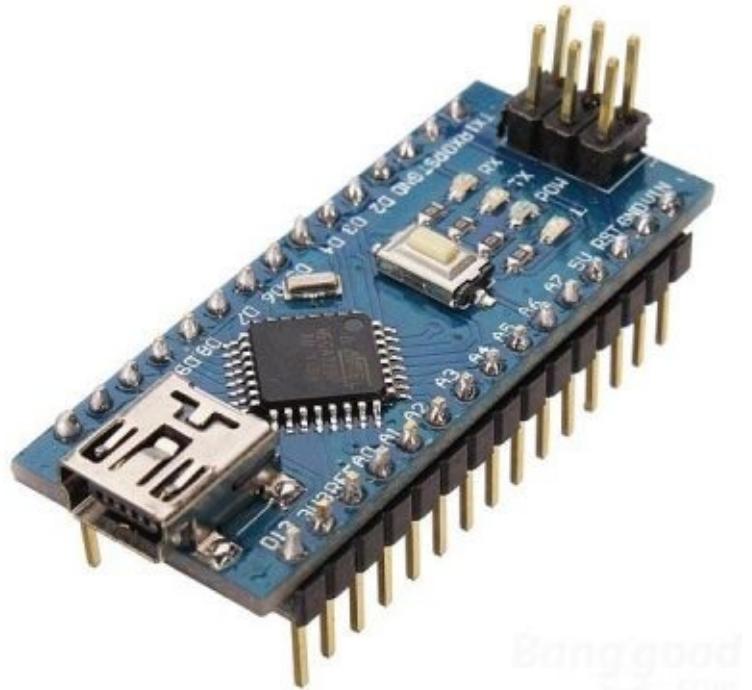
Unidades	Componentes
1	PCB Diseñada por XDeSIG
5	Resistencias cerámicas 10k 1/4W 5%
1	Resistencia cerámica 22k 1/4W 5%
5	Pulsadores 12x12x7.3 con carcasa de colores
3	Pines acodados

Tienes una [guía de soldadura](#), con cada uno de los pasos a seguir para ensamblar los componentes.

Arduino NANO

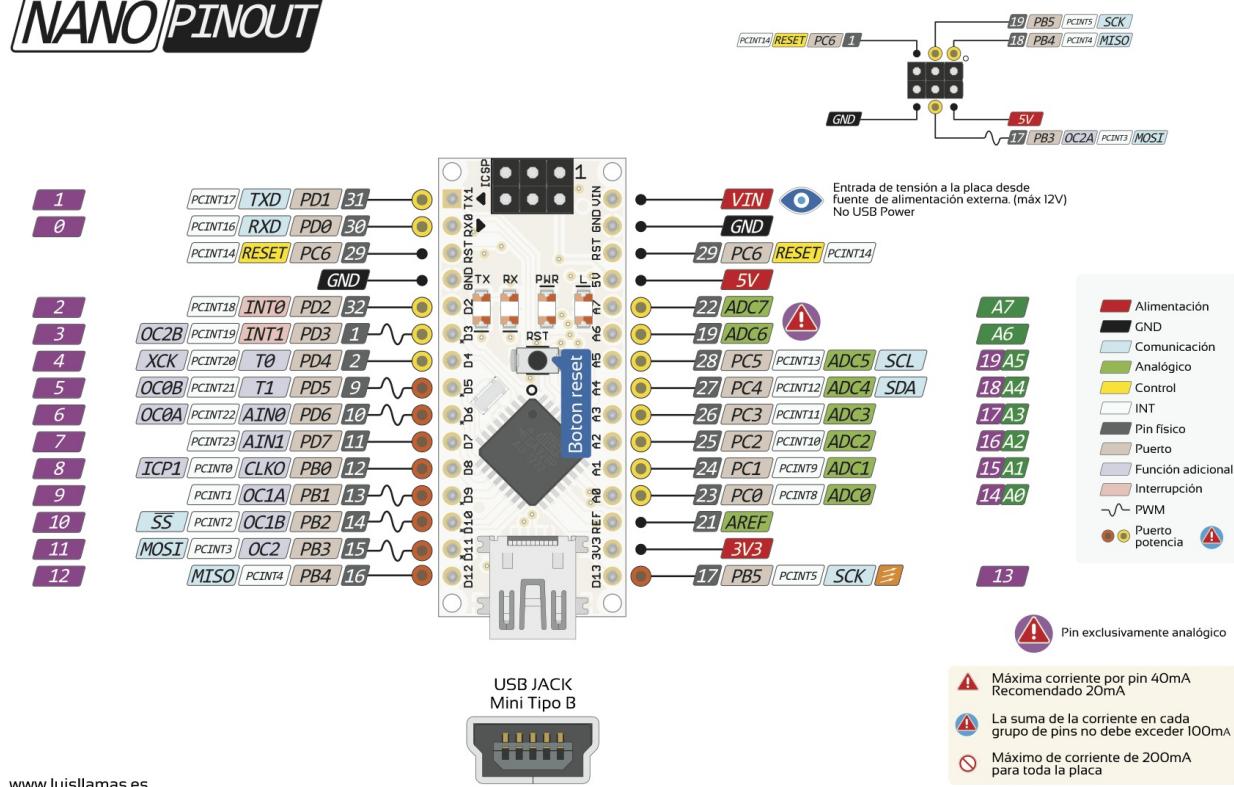
Es el **cerebro del robot** y donde [cargaremos nuestro código](#) a través de una conexión Micro o Mini-USB para conectarlo a nuestro ordenador.

Tiene dos filas de pinchos o patillas soldados en cada lateral y lo pondremos sobre la [protoboard](#).



Cada pincho esta conectado a un PIN que viene marcado con un nombre concreto y tiene una función concreta.. En esta imagen, sacada de la web de [Luis Llamas](#) puedes ver la correspondencia:

NANO PINOUT



¿Qué necesitas saber?

No hace falta comprender el funcionamiento completo de la placa, con entender porque vamos a utilizar los pines donde conectaremos Escornabot es suficiente. Si alguien tiene interés en adquirir conocimiento extra, durante el curso tendrá enlaces a páginas web donde poder conseguirlo.

Placa botonera

Utilizaremos tres cables (también llamados hilos), aunque con dos sería suficiente:

PIN Arduino NANO	¿Por qué?
5V	5V - Alimentar la placa botonera
A4	Sig - Transmitir las ordenes de los botones a la placa Arduino NANO para que luego esta de las ordenes a los motores.
GND	GND - Aunque no es correcto del todo lo llamaremos tierra o masa que es un término más conocido.

Porta Pilas

PIN Arduino NANO	¿Por qué?
VIN	El portapilas contiene 4 pilas de 1,5 V así que la entrada de corriente será de 6V y este es el único PIN que admite corriente desde 6V a 12V.
GND	Tierra o masa

Drivers ULN2003



PIN Arduino NANO	¿Por qué?
VIN	En esta placa van conectados los motores y necesitaremos conectarlo a este pin para poder coger corriente del Porta Pilas que conectaremos justo encima de los dos cables de corriente de los drivers.
GND	Tierra o masa
D9, D8, D7, D6 (en algunas placas 9, 8, 7, 6)	Conexión del motor izquierdo al Arduino Nano para poder recibir las ordenes de los movimientos y transmitirlas posteriormente al motor.
D5, D4, D3, D2 (en algunas placas 5, 4, 3, 2)	Conexión del motor derecho al Arduino Nano para poder recibir las ordenes de los movimientos y transmitirlas posteriormente al motor.

Buzzer 5V

PIN Arduino NANO	¿Por qué?
D10 (en algunas placas 10)	Para poder recibir las ordenes del Arduino NANO y ejecutar los pitidos según las pulsaciones de los botones.
GND	Tierra o masa

Motores paso a paso 28BY J-48

Es el componente que hace posible el desplazamiento del robot. Se mueven por pequeños pasos para avanzar en un sentido u otro.

Si quieres ampliar información, una [entrada de Prometec.net](#)

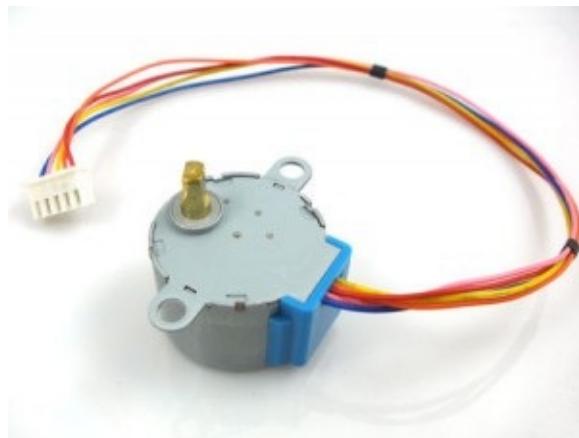


Imagen de [Prometec.net](#)

El conector blanco va colocado a un [Driver ULN2003](#) **del que recibirá tanto la alimentación como las órdenes**. Una comprobación simple para ver si el funcionamiento es correcto, girar la rueda y ver si se encienden los LEDs de la parte inferior en los Drivers.

Un problema que puede darnos dolores de cabeza es tener un **cableado de motores incorrecto**. Nos daremos cuenta porque teniendo todo en su sitio y la programación bien, nuestro escorramabot invierte algunos movimientos de la botonera. El robot se diseñó y programó para funcionar con los motores conectados de una manera concreta, lo tienes explicado en [esta entrada](#).

Drivers ULN2003

En el **conector blanco** de la placa pondremos **los motores**. Cada motor debe ir conectado al **driver de su lado** o el robot se moverá al contrario de las órdenes que le demos.

Encima del conector blanco tenemos **cuatro pines o pinchos** que son los que **conectaremos a los pines D9-D2** de la placa Arduino NANO.

A la izquierda tenemos **dos pines o pinchos** que son los encargados de recibir la alimentación del portapilas, están **marcados con la serigrafía 5-12V y +/-**



Recuerda el cuadro de conexiones:

PIN Arduino NANO	PIN Driver	¿Por qué?
VIN	5-12V (+)	En esta placa van conectados los motores y necesitaremos conectarlo a este pin para poder coger corriente del Porta Pilas que conectaremos justo encima de los dos cables de corriente de los drivers.
GND	5-12V (-)	Tierra o masa
D9, D8, D7, D6 (en algunas placas 9, 8, 7, 6)	IN1, IN2, IN3, IN4	Conexión del motor izquierdo al Arduino Nano para poder recibir las órdenes de los movimientos y transmitirlas posteriormente al motor.
D5, D4, D3, D2 (en algunas placas 5, 4, 3, 2)	IN1, IN2, IN3, IN4	Conexión del motor derecho al Arduino Nano para poder recibir las órdenes de los movimientos y transmitirlas posteriormente al motor.

Protopboard 170 puntos

En esta placa **colocaremos el Arduino NANO** y nos servirá de enlace con el resto de componentes.



El funcionamiento es simple, **sirve para ampliar las conexiones a los pines** de la placa Arduino.

Los huecos que queden encima o debajo de cada pin son una extensión del mismo, con una imagen se entiende mejor.



Siquieres ampliar información, una [entrada de rinconingenieril.es](http://rinconingenieril.es)

Buzzer 5V

Es el altavoz que dota de sonido a Escornabot, cada vez que pulsemos una tecla emitirá un pitido.

Tiene una **patilla larga o positivo** y una **patilla corta o negativo**. Si las dos patillas tienen la misma longitud en la parte delantera tienen serigrafiado el valor positivo.

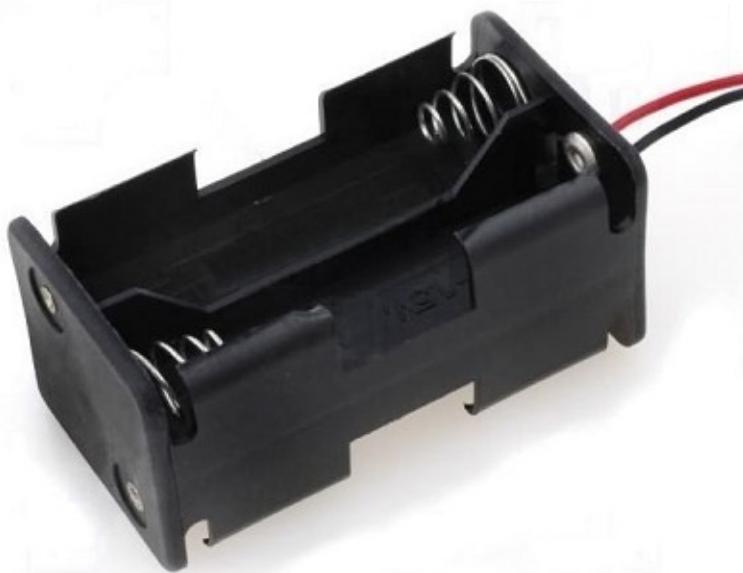


Recuerda el cuadro de conexiones:

PIN Arduino NANO	Buzzer 5V	¿Por qué?
D10 (en algunas placas 10)	Patilla larga (+)	Para poder recibir las ordenes del Arduino NANO y ejecutar los pitidos según las pulsaciones de los botones.
GND	Patilla corta (-)	Tierra o masa

Portapilas 4 unidades AA

El portapilas **proporciona la energía para el movimiento** del robot.



Lleva **4 pilas del tipo AA**, hay que fijarse al introducir las pilas para hacerlo en el sentido correcto. Parece algo evidente pero no sería la primera, ni la última vez que una persona las coloca mal y se coge un buen calentón.

Una regla que suelo usar con los peques:

- La parte plana de la pila, va siempre **en el lado del muelle**
- La parte de la pila que tiene un saliente, en el otro lado.

Recuerda el cuadro de conexiones:

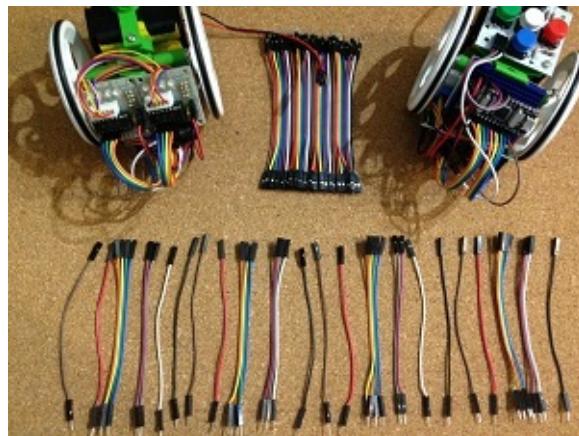
PIN Arduino NANO	Portapilas	¿Por qué?
VIN	Cable rojo (+)	El portapilas contiene 4 pilas de 1,5 V así que la entrada de corriente será de 6V y este es el único PIN que admite corriente desde 6V a 12V .
GND	Cable negro (-)	Tierra o masa

La terminación de los cables se puede [crimpar para tener un conector macho](#) en el extremo.

Cables Arduino macho-hembra

Se llaman **cables dupont** pero me gusta llamarlos cables Arduino porque así cualquier persona lo entiende. **Utilizaremos cables macho-hembra**, que son los que tienen el pincho en un lado y el hueco en el otro.

Vienen en tiras de 40 y esta es la forma en la que me gusta separarlos:



- Marrón
- Rojo
- Cuatro colores juntos; azul, verde, amarillo y naranja
- Dos colores juntos; violeta y gris
- Blanco
- Negro
- Marrón
- Rojo
- Cuatro colores juntos; azul, verde, amarillo y naranja
- Tres colores juntos; violeta, gris y blanco
- Negro

Con estos cables realizaremos la conexión entre los diferentes componentes, un par de consejos:

- Si ves que **el cable no entra en la protoboard**, no fuerces. **Gira 90 grados** antes de volver a intentarlo.
- Cuando el **conector se dobla y queda frágil cámbialo por uno nuevo**, es un incordio si se parte dentro de la **protoboard**.
- Si se parte un conector dentro de la protoboard, con unas pinzas de punta fina puedes retirarlo.

Canica 14 mm

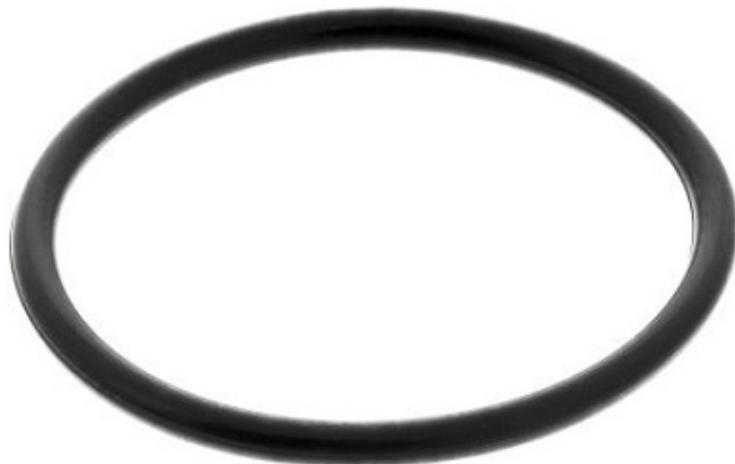
Nos **indica la parte trasera del robot**, hace de rueda loca y aporta estabilidad para los giros.



Si vemos que se sale de la pieza impresa, podemos calentar con mucho cuidado la pieza y moldearla un poco para que la canica quede fija.

Juntas tóricas 63x57x3 mm

Se colocan alrededor de las ruedas a modo de cubiertas como en las ruedas de las bicicletas. **Aportan agarre y estabilidad** para que los **movimientos del robot sean más exactos.**



Tornillería y tuercas

Sirven para ensamblar los componentes a las piezas impresas en 3D.

Tornillería

Se utilizan **16 tornillos de métrica 3 y 10mm** de longitud.

Para componentes como la [placa botonera](#) o los [drivers ULN2003](#) utilice **tornillos de 5mm** (la mitad de cortos) porque facilitan el montaje



También se pueden utilizar tornillos perforantes, el problema de estos es que si desmontas y montas varias veces el mismo robot, dejan de hacer su función. Son más agresivos a la hora de atornillarse en el plástico, si quedan sueltos utiliza uno de los anteriores.



Tuerca

Se utilizan **2 tuercas de métrica 3** para introducir una en cada rueda y así poderlas fijar a los motores.



Guía de montaje

Puedes consultar y descargar la [guía de montaje aquí](#)



Instalar programación

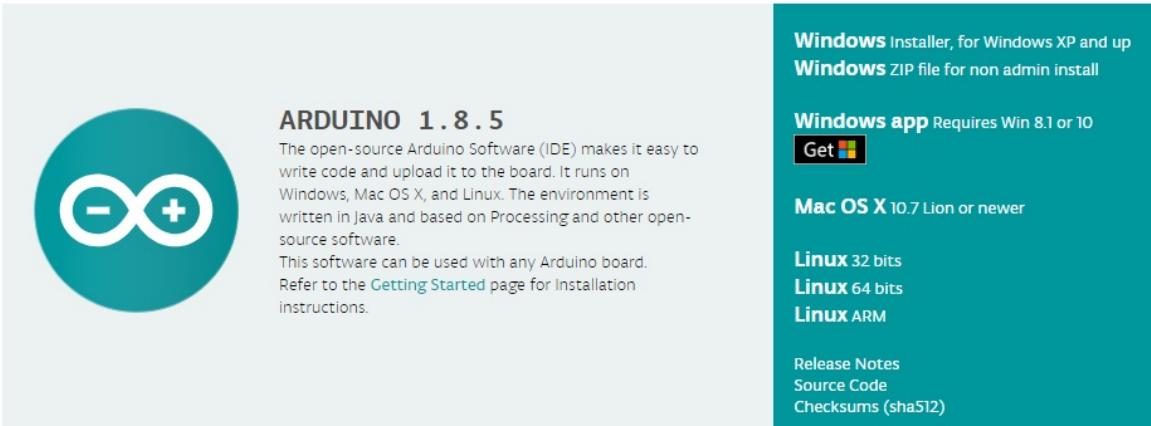
No te preocupes si es la primera vez que utilizas Arduino, vamos a ver paso a paso como se instala el programa y se carga la programación en nuestra placa [Arduino Nano](#)

Para que sea más agradable, todo el proceso se ha dividido en pequeños hitos:

- [Descargando IDE Arduino](#)
- [Instalando IDE Arduino](#)
- [Abriendo IDE Arduino](#)
- [Descargando la configuración para Escornabot](#)
- [Test de botonera](#)
- [Cambiar los valores de la botonera](#)
- [Cargar código modificado](#)

Descargando IDE Arduino

Primero entramos en la [sección descargas](#) de su web:



The screenshot shows the Arduino website's download section. At the top, there's a navigation bar with links for HOME, BUY, SOFTWARE (which is highlighted in blue), PRODUCTS, EDU, RESOURCES, COMMUNITY, and HELP. Below the navigation is a search icon. The main heading is "Download Arduino IDE". Underneath, there's a large image of the Arduino logo (an infinity symbol with a minus and plus sign) and a brief description of the Arduino IDE. To the right, there are download links for Windows (Windows Installer and ZIP file), Windows app (with a "Get" button), Mac OS X (Mac OS X 10.7 Lion or newer), Linux (32 bits, 64 bits, ARM), and Release Notes, Source Code, Checksums (sha512). A red box highlights the "DOWNLOADS" button in the top navigation.

En la parte derecha tienes que elegir la versión para tu sistema operativo, en este caso he utilizado la versión 1.8.5. Es **importante** descargar una versión testeada y no una beta que puede darte problemas al hora de identificar o subir el código.

Después nos sale la página por si queremos hacer una donación al proyecto Arduino o descargar. En la imagen tienes marcada la opción descargar directamente:

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)



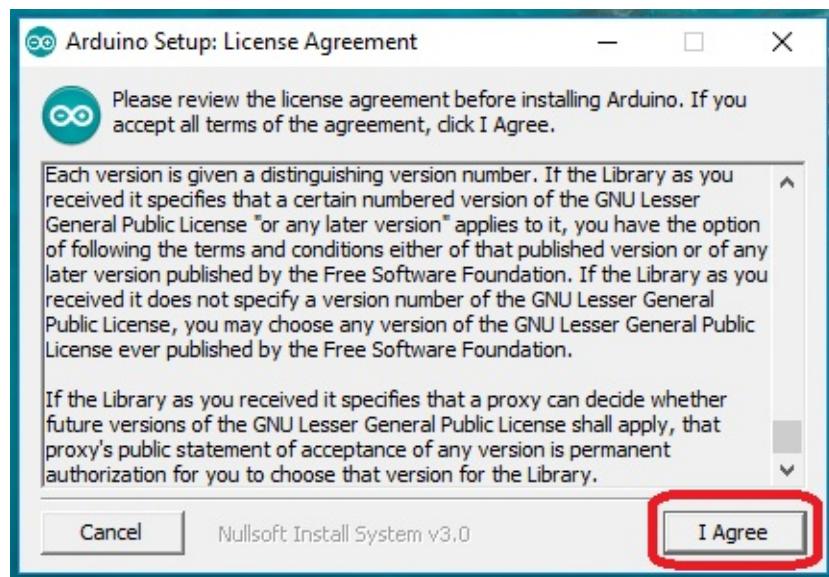
SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED 24,338,475 TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!

\$3 \$5 \$10 \$25 \$50 OTHER

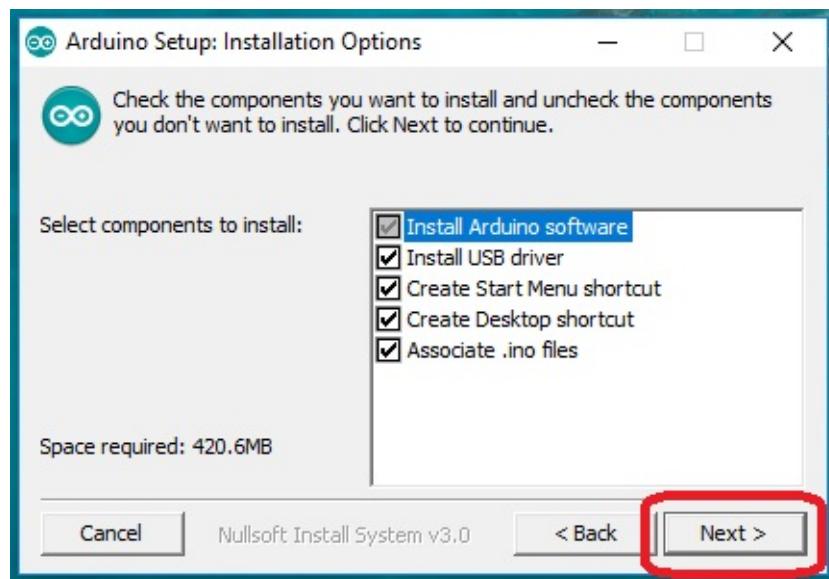
JUST DOWNLOAD CONTRIBUTE & DOWNLOAD

Instalando IDE Arduino

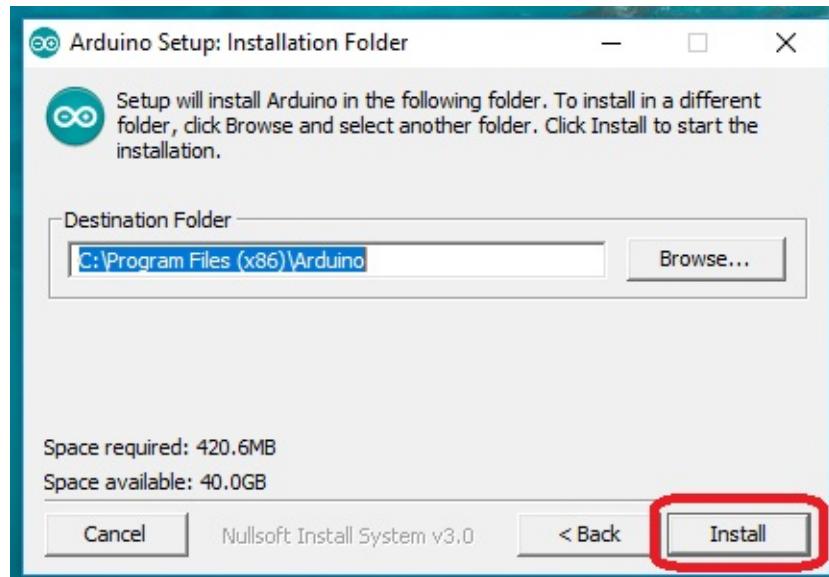
Vamos a la carpeta de descargas en nuestro PC y ejecutamos el programa de instalación. Se nos abre la pantalla de instalación, lo primero aceptar las licencias del programa:



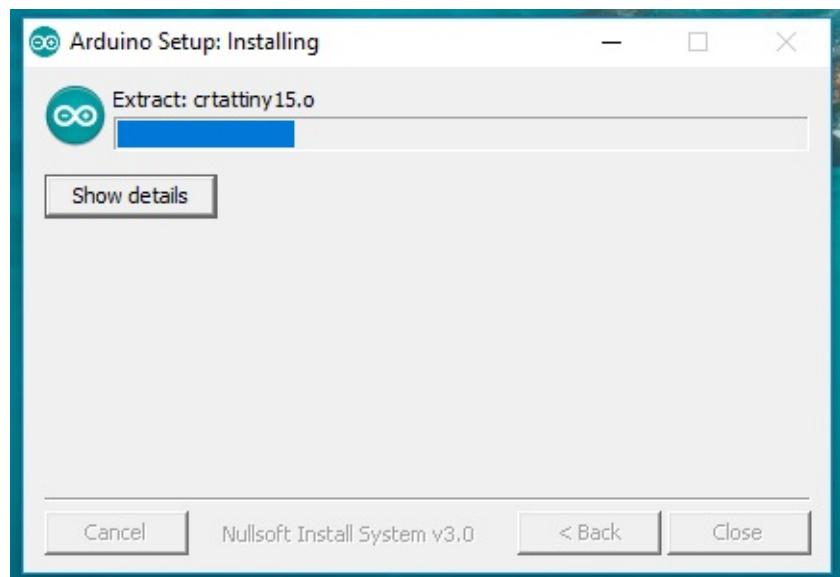
En la siguiente pantalla seleccionamos los componentes que queremos instalar (seleccionar todo):



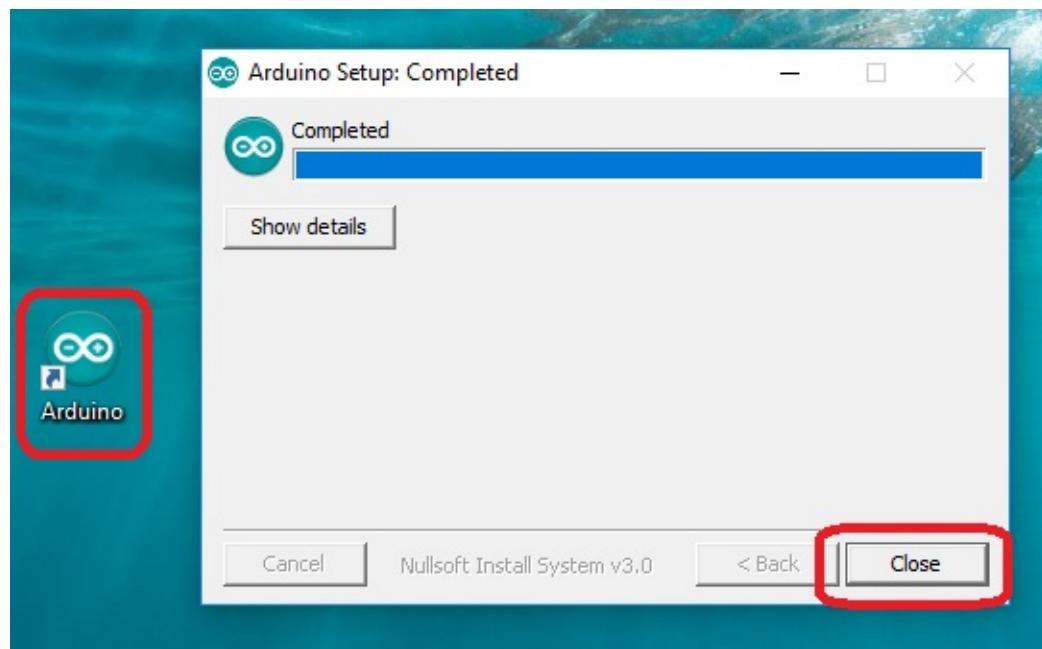
Después seleccionamos la carpeta de nuestro ordenador donde queremos realizar la instalación, es importante apuntar esta dirección porque tarde o temprano necesitaremos entrar en esta carpeta.



Pulsamos instalar y esperamos:



Una vez completada, cerramos la ventana de instalación y en el escritorio tendremos el acceso directo al programa:



Abriendo IDE Arduino

Ya **tenemos Arduino instalado en nuestro PC**, ahora si pinchamos en el icono creado en el escritorio se abrirá la página principal del programa.



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_jun10a Arduino 1.8.5". Below it is a menu bar with "Archivo", "Editar", "Programa", "Herramientas", and "Ayuda". The main area displays the following code:

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:
}
```

Os dejo un [enlace con varias web](#) donde podeis encontrar manuales y tutoriales sobre Arduino para quienes quieran ampliar sus conocimientos.

En este curso **aprenderemos lo necesario para que nuestro escornabot funcione**.

Descargando la configuración para Escornabot

Rafa Couto es el creador del código y tiene un [repositorio con todas las versiones](#). Utilizaremos para este ejemplo la versión 1.4.3 pero puedes elegir la que quieras.

Latest release v1.4.3 · rafacouto · 3cf1ae2

Assets

- Source code (zip)
- Source code (tar.gz)

Fixed bugs:

- 3-wires keypad not working correctly (reported by Pedro García Pombo).

Descargamos el archivo .zip y lo descomprimimos en nuestro PC, tendremos una carpeta con los siguientes archivos:

equipo > Descargas > arduino-1.4.3 >

Nombre
config
Escornabot
.gitignore
.travis.yml
CHANGELOG.md
LICENSE
platformio.ini
README.md

En la carpeta "Escornabot" tenemos la configuración que usaremos para cargar en nuestro robot:

Recuerda, debes abrir el archivo "Escornabot.ino"

en equipo > Descargas > arduino-1.4.3 > Escornabot

Nombre	Fecha de i
Buzzer.h	14/12/201
Configuration.h	31/05/201
Engine.cpp	14/12/201
Engine.h	14/12/201
EngineSteppers.cpp	14/12/201
EngineSteppers.h	14/12/201
Enums.h	14/12/201
Escornabot.h	14/12/201
Escornabot.ino	14/12/201
EventHistory.h	14/12/201
EventManager.cpp	14/12/201
EventManager.h	14/12/201
KeypadLeds.cpp	14/12/201
KeypadLeds.h	14/12/201
MoveList.cpp	14/12/201
MoveList.h	14/12/201
PersistentMemory.cpp	14/12/201

Test de botonera

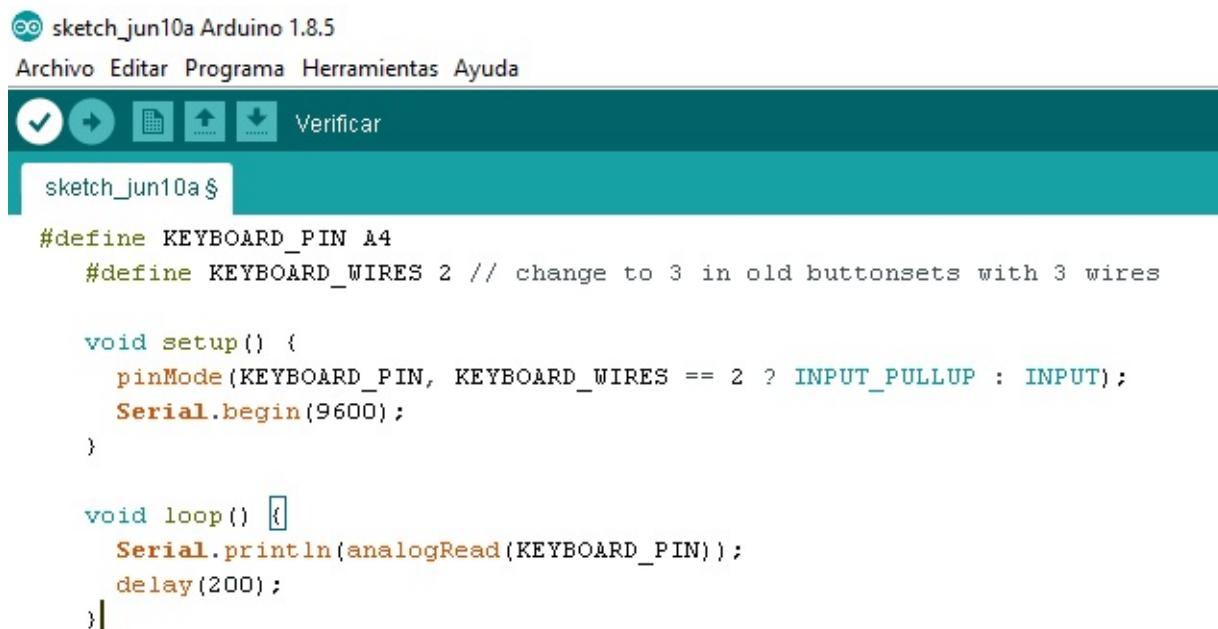
Antes de seguir es muy importante que aprendas a realizar el test de botonera. Lo tienes perfectamente explicado en este enlace de la web oficial [escornabot](#).

- Abre arduino, un nuevo proyecto y borra el código que aparezca.
- Pega el siguiente código:

```
#define KEYBOARD_PIN A4
#define KEYBOARD_WIRES 2 // change to 3 in old buttonsets with 3 wires

void setup() {
    pinMode(KEYBOARD_PIN, KEYBOARD_WIRES == 2 ? INPUT_PULLUP : INPUT);
    Serial.begin(9600);
}

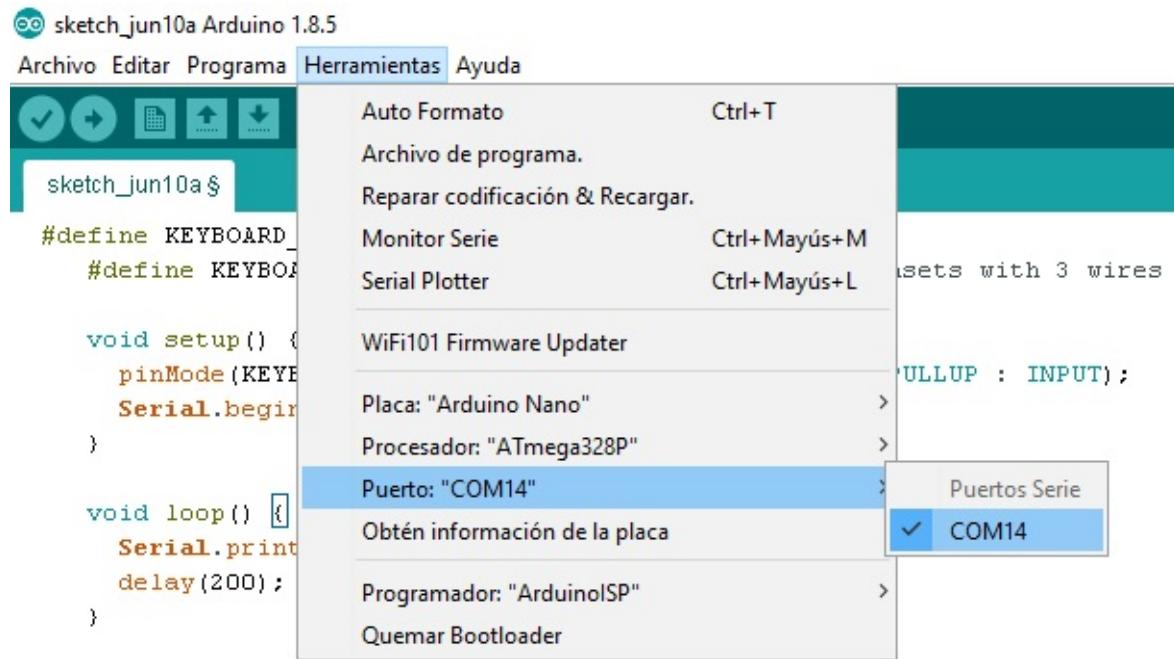
void loop() {
    Serial.println(analogRead(KEYBOARD_PIN));
    delay(200);
}
```



- Conecta el Arduino Nano al ordenador mediante un cable USB.
- En la pestaña herramientas **selecciona como placa "Arduino Nano"**.
- En la pestaña herramientas **selecciona como procesador "Atmega328P"**.
- En la pestaña herramientas **selecciona el puerto del PC** al que lo has conectado, en mi caso "COM14".

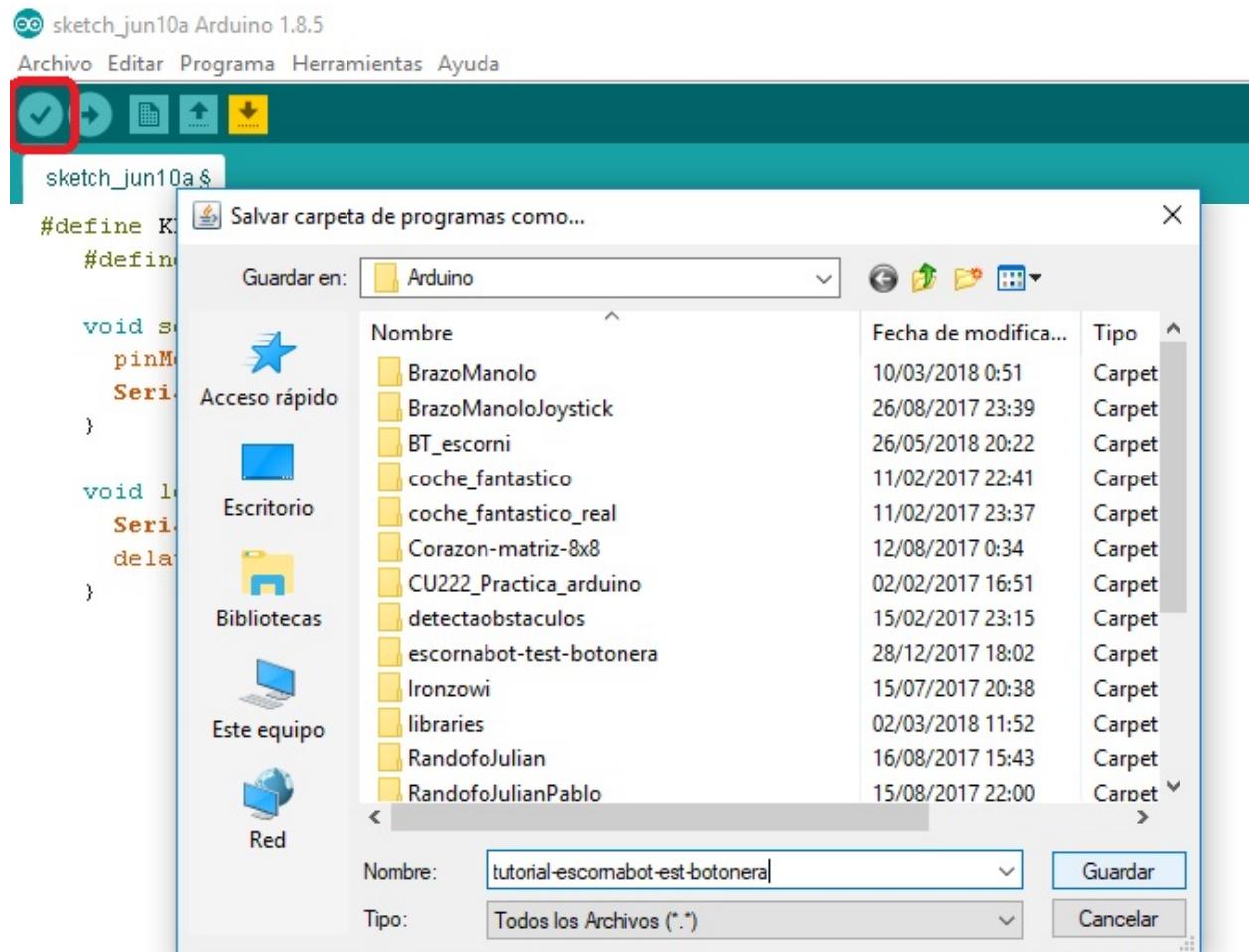
En las últimas versiones del IDE Arduino hay un cambio que afecta a las placas que utilizamos y **debemos seleccionar en procesador; "ATmega328P (Old Bootloader)"**

Toda la información en [esta entrada](#)



Ya tenemos todo listo para cargar el código.

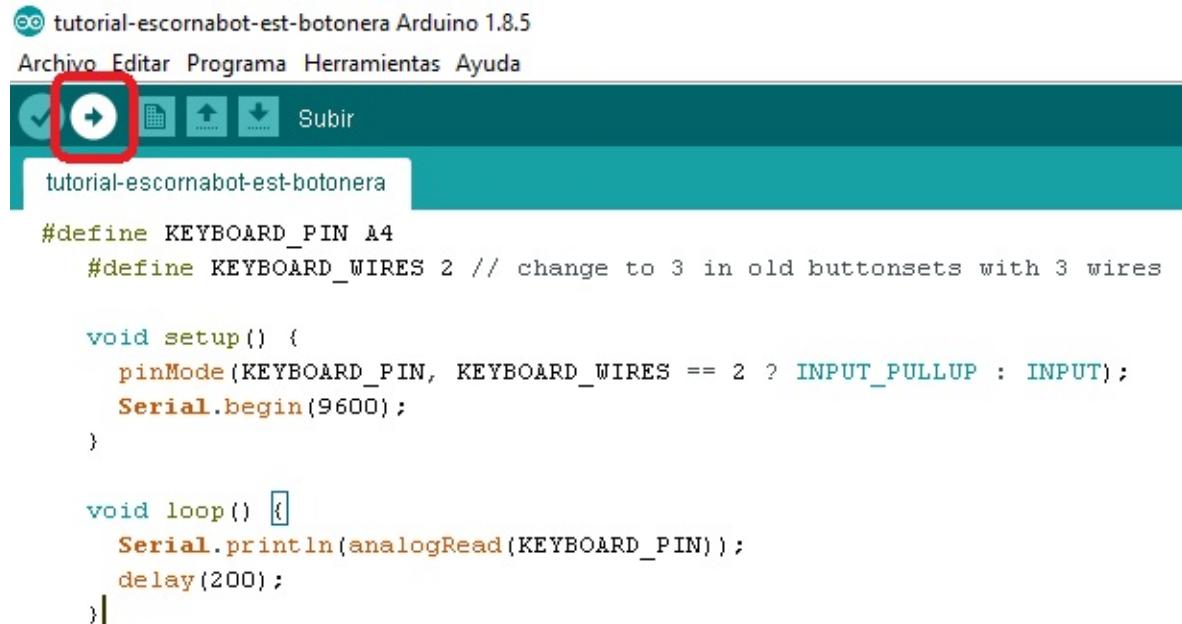
- Primero le **damos al botón de verificar**, lo tienes en la **parte superior izquierda con un símbolo de una "V"**, al pulsar se abrirá una ventana para que guardes el proyecto con el nombre que te apetezca:



En la parte inferior nos tiene que aparecer el mensaje en Azul de compilado.

```
Compilado
Archiving built core (caching) in:
El Sketch usa 2054 bytes (6%) del espacio
Las variables Globales usan 188 bytes (9%)
```

- Con este paso hemos confirmado que el código lo tenemos copiado de manera correcta.
- Ahora **vamos a subir el código** a nuestra placa, para eso **pulsaremos en el botón con forma de flecha** que tenemos al lado de la barra de herramientas:

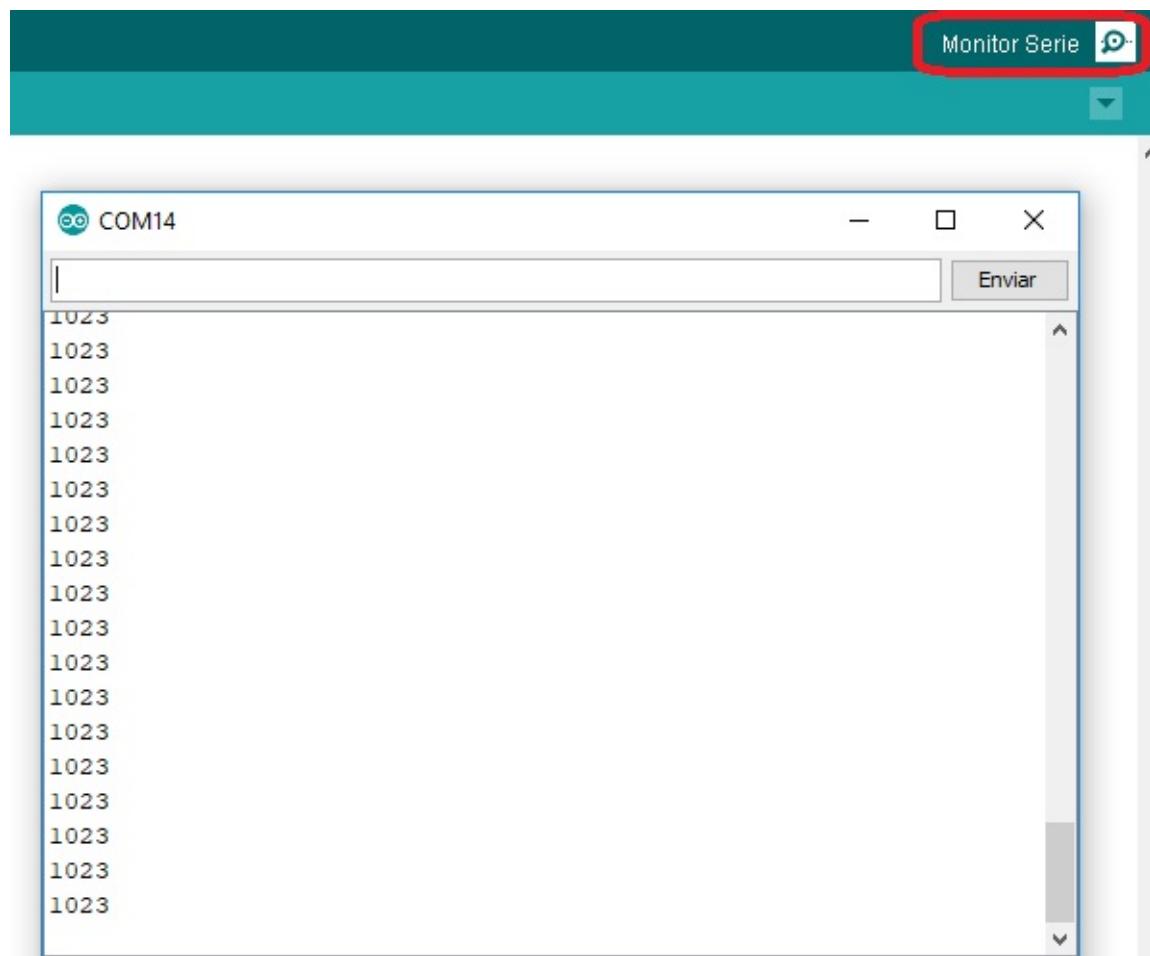


- Si todo ha ido bien en la parte inferior nos **tiene que aparecer en azul "Subido"** y ningún mensaje de error:

```
Subido
El Sketch usa 2054 bytes (6%) del espacio
Las variables Globales usan 188 bytes (9%)
```

- Ahora **abrimos el monitor serie** pulsando en la lupa que hay en la parte superior derecha y nos aparece una ventana con un valor en torno a 1023. **Cada vez que dejemos pulsado un botón este valor cambiará**, anota el valor de cada botón.

Recuerda, la parte delantera del robot son los motores y el cableado. La parte trasera, la bola.



Cambiando los valores de la botonera en la configuración

Ya lo hemos visto antes, vamos a la carpeta que hemos descargado con la configuración y entramos dentro de la **carpeta "Escornabot"** para abrir el archivo "**Escornabot.ino**"

Nombre	Fecha de
Buzzer.h	14/12/201
Configuration.h	31/05/201
Engine.cpp	14/12/201
Engine.h	14/12/201
EngineSteppers.cpp	14/12/201
EngineSteppers.h	14/12/201
Enums.h	14/12/201
Escornabot.h	14/12/201
Escornabot.ino	14/12/201
EventManager.h	14/12/201
EventManager.cpp	14/12/201
KeypadLeds.cpp	14/12/201
KeypadLeds.h	14/12/201
MoveList.cpp	14/12/201
MoveList.h	14/12/201
PersistentMemory.cpp	14/12/201

Tienes que ver todas estas pestañas superiores. De lo contrario, algo no hiciste bien:

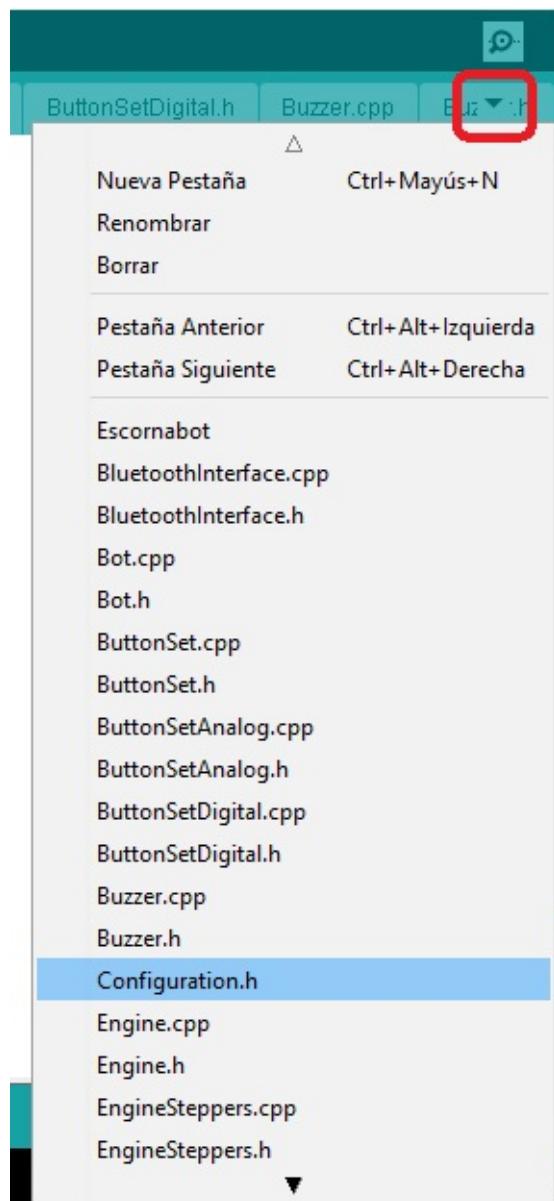
```
// Escornabot.ino
/*
Copyright (C) 2014-2016 Escornabot - http://escornabot.com

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301,
USA.
*/
```

Entramos en la pestaña "configuration.h" para eso pulsamos en el icono pequeño de la parte superior derecha del a imagen y se abre un menú desplegable:



Buscamos el apartado donde nos aparecen los valores de la botonera y los cambiamos por los que hemos anotado:

Escornabot - Configuration.h | Arduino 1.8.5

Archivo Editar Programa Herramientas Ayuda

```
Escornabot BluetoothInterface.cpp BluetoothInterface.h Bot.cpp Both Button.h
////////////////////////////////////////////////////////////////
////// Button set analog
////////////////////////////////////////////////////////////////

#ifndef BUTTONS_ANALOG
#define BS_ANALOG_WIRES 2
//#define BS_ANALOG_WIRES 3

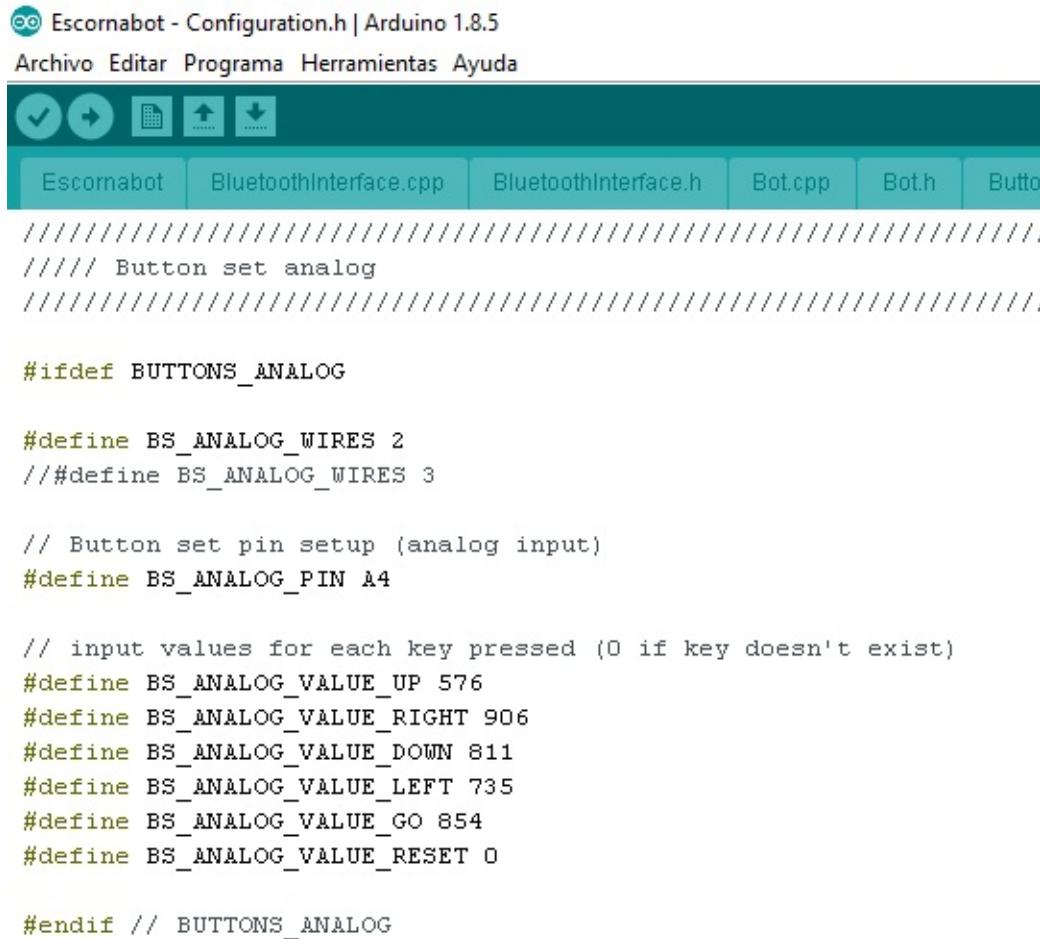
// Button set pin setup (analog input)
#define BS_ANALOG_PIN A4

// input values for each key pressed (0 if key doesn't exist)
#define BS_ANALOG_VALUE_UP 576
#define BS_ANALOG_VALUE_RIGHT 906
#define BS_ANALOG_VALUE_DOWN 811
#define BS_ANALOG_VALUE_LEFT 735
#define BS_ANALOG_VALUE_GO 854
#define BS_ANALOG_VALUE_RESET 0

#endif // BUTTONS_ANALOG
```

Cargar el código modificado

Ya tenemos el código con los valores de nuestra botonera modificados, **nos aseguramos** también que la **configuración del teclado esta en el pin A4** (En la imagen lo puedes ver).



```

//////////////////////////////////////////////////////////////////
// Button set analog
//////////////////////////////////////////////////////////////////

#ifndef BUTTONS_ANALOG
#define BS_ANALOG_WIRES 2
// #define BS_ANALOG_WIRES 3

// Button set pin setup (analog input)
#define BS_ANALOG_PIN A4

// input values for each key pressed (0 if key doesn't exist)
#define BS_ANALOG_VALUE_UP 576
#define BS_ANALOG_VALUE_RIGHT 906
#define BS_ANALOG_VALUE_DOWN 811
#define BS_ANALOG_VALUE_LEFT 735
#define BS_ANALOG_VALUE_GO 854
#define BS_ANALOG_VALUE_RESET 0

#endif // BUTTONS_ANALOG

```

Realizamos las mismas operaciones que antes:

- Pulsando en la "V" **verificamos** el código.
- Pulsando en la "flecha" **subimos** el código.

Si todo ha ido bien, **el robot emitirá un pitido del check inicial** indicando que todo esta correcto.

Pulsamos uno a uno cada botón ejecutando la acción de manera individual para comprobar su funcionamiento y ya podemos jugar con nuestro Escornabot.

Revisa el apartado [Test de botonera](#), si tienes algún problema al cargar el código.

Modificar Programación

Tenemos el **robot montado**, la **programación instalada** y todo **funciona correctamente**.

Las siguientes preguntas que surgen son:

- ¿Cómo hago que vaya más deprisa?
- ¿Cómo modifco la distancia que recorre?
- ¿Cómo modifco los giros?

A pesar de parecer una programación muy compleja, es fácil realizar los cambios en el código y lo vais a ver paso a paso:

- [Modificar Velocidad](#)
- [Modificar Distancia de avance](#)
- [Modificar Grados de giro](#)

Modificar Velocidad

Ya aprendimos en el apartado [cambiar los valores de la botonera](#) como **entrar en la pestaña "Configuration.h"** de la programación.

Abrimos esa pestaña y localizamos la siguiente línea:

```
#define STEPPERS_STEPS_PER_SECOND 1000

///////////////////////////////
////// Steppers engine setup
///////////////////////////////

#ifndef ENGINE_TYPE_STEPPERS

// stepper pin setup (digital outputs)
#define STEPPERS_MOTOR_RIGHT_IN1 5
#define STEPPERS_MOTOR_RIGHT_IN2 4
#define STEPPERS_MOTOR_RIGHT_IN3 3
#define STEPPERS_MOTOR_RIGHT_IN4 2
#define STEPPERS_MOTOR_LEFT_IN1 9
#define STEPPERS_MOTOR_LEFT_IN2 8
#define STEPPERS_MOTOR_LEFT_IN3 7
#define STEPPERS_MOTOR_LEFT_IN4 6

// step calibration
#define STEPPERS_STEPS_PER_SECOND 1000
#define STEPPERS_LINE_STEPS 1738
#define STEPPERS_TURN_STEPS 1024

#endif
```

Esta línea nos define los pasos por segundo que va a realizar el motor cada vez que pulsemos en el botón de ir adelante o atrás. **Si aumentamos el valor, aumentará la velocidad** y si disminuimos el valor, disminuye la velocidad.

El límite está en torno a los 2300 steps, dependerá del voltaje de las pilas que el valor sea mayor o menor.

Juego - Dejar que los peques busquen por ellos mismos el límite del valor de los steps de su robot.

Modificar Velocidad de avance

Este apartado además de servir para trabajar medidas, reglas de tres o distancias. Nos permite modificar el avance para jugar en tableros que ya tengamos de otros robots con casillas de 15 cm.

Con la programación abierta y situados en la pestaña "Configuration.h" localizamos la siguiente línea:

```
#define STEPPERS_LINE_STEPS 1738

///////////////////////////////
//// Steppers engine setup
///////////////////////////////

#ifndef ENGINE_TYPE_STEPPERS

// stepper pin setup (digital outputs)
#define STEPPERS_MOTOR_RIGHT_IN1 5
#define STEPPERS_MOTOR_RIGHT_IN2 4
#define STEPPERS_MOTOR_RIGHT_IN3 3
#define STEPPERS_MOTOR_RIGHT_IN4 2
#define STEPPERS_MOTOR_LEFT_IN1 9
#define STEPPERS_MOTOR_LEFT_IN2 8
#define STEPPERS_MOTOR_LEFT_IN3 7
#define STEPPERS_MOTOR_LEFT_IN4 6

// step calibration
#define STEPPERS_STEPS_PER_SECOND 1000
#define STEPPERS_LINE_STEPS 1738
#define STEPPERS_TURN_STEPS 1024

#endif
```

El **valor 1738 equivale a un avance de 10cm** cada vez que pulsamos los botones para desplazarnos adelante o atrás.

Realizando una regla de tres simple, podemos ver que **el valor de avance de 1cm será de 174** (se redondea el valor porque los decimales no se van a tener en cuenta)

Sabiendo el avance de un centímetro podremos cambiar el valor para que desplace la distancia que queramos.

Juego - Practicar reglas de tres para que calculen diferentes distancias con marcas en el suelo que deben ir alcanzando.

Modificar Grados de giro

Es el turno de los giros, con la programación abierta y situados en la pestaña "Configuration.h" localizamos la siguiente línea:

```
#define STEPPERS_TURN_STEPS 1024

////////// Steppers engine setup
////////// Steppers engine setup

#ifndef ENGINE_TYPE_STEPPERS

// stepper pin setup (digital outputs)
#define STEPPERS_MOTOR_RIGHT_IN1 5
#define STEPPERS_MOTOR_RIGHT_IN2 4
#define STEPPERS_MOTOR_RIGHT_IN3 3
#define STEPPERS_MOTOR_RIGHT_IN4 2
#define STEPPERS_MOTOR_LEFT_IN1 9
#define STEPPERS_MOTOR_LEFT_IN2 8
#define STEPPERS_MOTOR_LEFT_IN3 7
#define STEPPERS_MOTOR_LEFT_IN4 6

// step calibration
#define STEPPERS_STEPS_PER_SECOND 1000
#define STEPPERS_LINE_STEPS 1738
#define STEPPERS_TURN_STEPS 1024

#endif
```

Esta línea nos dice cada vez que pulsamos los botones derecha o izquierda, cuantos grados girará Escornabot.

El valor **1024 indica giros de 90 grados.**

Al igual que en el apartado anterior, si realizamos una regla de tres simple podemos saber el valor gire de otra manera.

Recordar, en caso de tener decimales hay que redondear.

Por ejemplo, si queremos realizar giros de 45 grados, el valor equivalente será de 512.

Juego - Practicar las horas del reloj variando los ángulos de giro. Pueden jugar por parejas con dos esferas, una para marcar los minutos y otra para marcar los segundos.

Material para trabajar en clase

En este apartado iremos recopilando todas las **actividades y materiales** de utilidad para trabajar en clase.

Es **importante documentar las actividades**, compartir y hacerlas accesible para que cualquier persona, en cualquier parte el mundo tenga la posibilidad de acceder a ellas.

- [Tableros y fichas](#)
- [Disfraces](#)
- [Máscaras Impresión 3D](#)
- [Dados CoDices](#)

Diseños ya hechos

En [thingiverse](#) podéis encontrar muchos diseños para descargar e imprimir directamente.

Tableros y fichas

Estos son algunos ejemplos, el material completo y actualizado lo puedes consultar en la [sección tableros y recursos de mi página web](#).

Estamos realizando un recopilatorio con actividades a través de [este formulario](#)

Puedes consultar las [aportaciones actuales aquí](#)

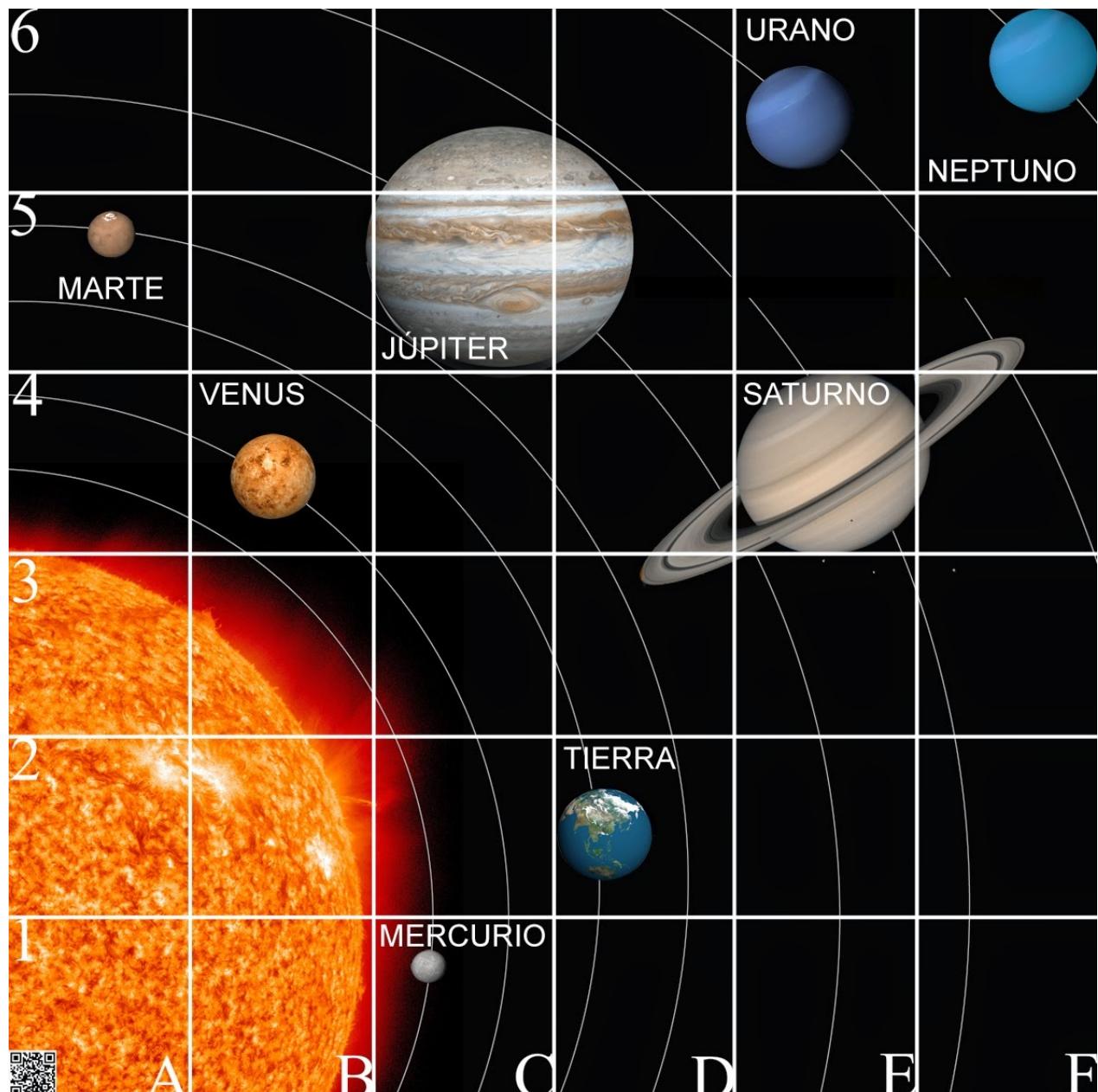
Tableros

Tableros con actividades

[Recopilatorio con tableros y actividades](#) en el repositorio oficial escornabot realizado por [Jorge Lobo](#)

Tablero del Sistema Solar

Lo tienes explicado por [Jorge Lobo](#) en su blog [El Sistema Solar](#)

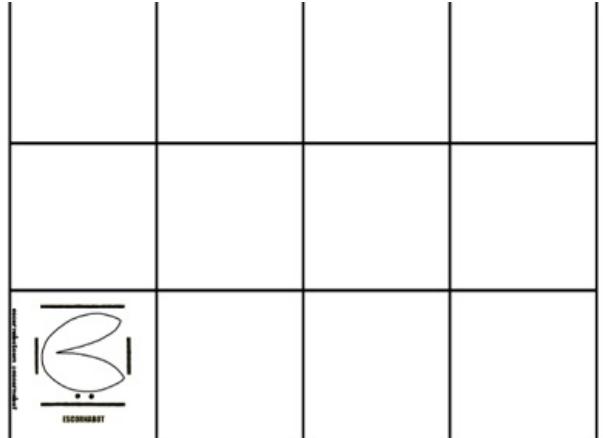
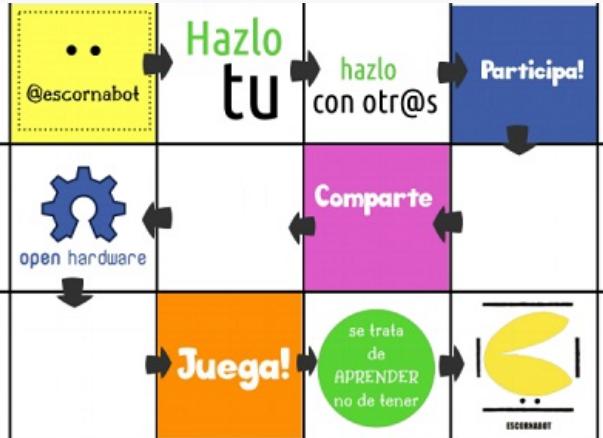


Tablero casillas blancas y oficial escornabot

El tablero con casillas blancas es muy interesante si le ponemos bolsas tipo zip cortadas en cada casilla y las utilizamos para meter tarjetas personalizadas.

Tienes en la parte inferior las tarjetas con recorridos, recortadas van eligiendo al azar un recorrido.

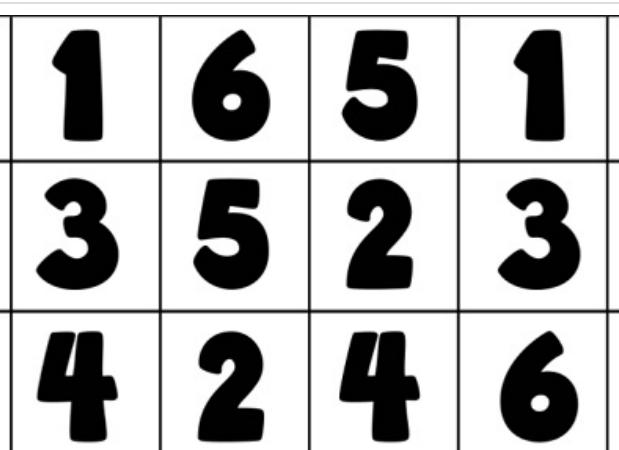
Tablero A-3	Archivo SVG	Archivo PDF

	Plantilla cuadrados formato .SVG	Plantilla cuadrados formato .PDF
	Tablero folleto formato .SVG	Tablero folleto formato .PDF

Tablero números

Puedes jugar tirando un dado y que busquen la casilla, realizando sumas simples... Edita el tablero para complicarlo.

Otro juego divertido son las carreras. Es para dos personas, cada uno con un escornabot y fuera del tablero. Se lanza un dado, como los números se repiten el objetivo es que alcancen la primera casilla que vean.

Tablero A-3	Archivo SVG	Archivo PDF
	Tablero Números formato .SVG	Tablero Números formato .PDF

Tablero condicional

Se debe llegar desde la casilla inferior izquierda con el triángulo verde hasta la casilla inferior derecha cumpliendo las condiciones de las casillas. La casilla con el aspa roja no se puede pisar, del mismo modo que la azul ya que se caería al agua.

Tablero A-3	Archivo SVG	Archivo PDF

	Tablero condicional formato .SVG	Tablero condicional formato .PDF
--	--	--

Tablero figuras geométricas

Se pueden recortar las figuras interiores o cambiar de color para que l@s peques primero hagan la composición del tablero, luego juegan combinando colores y formas geométricas. Puedes crear dos dados, uno contiene colores y otro las formas geométricas.

Tablero A-3	Archivo SVG	Archivo PDF
	Tablero-Cuadrado-Triangulo-Circulo.svg	Tablero-Cuadrado-Triangulo-Circulo.pdf

Tableros alfabeto

Para trabajar las letras con animalicos :)

Tablero A-3	Archivo SVG	Archivo PDF
	tablero-	tablero-

BOSQUEDEFANTASIAS.COM alfabeto infantil	A  EL Águila	B  EL BISONTE	C  EL CEBRA
D  EL DELFIN	E  EL ELEFANTE	F  LA FOCA	juega
 ESCORNABOT	G  EL GALLO	H  EL HIPOPÓTAMO	I  LA IGUANA

BOSQUEDEFANTASIAS.COM alfabeto infantil	J  LA JIRAFÁ	K  EL KOALA	L  EL LEÓN
M  EL MONO	N  LA NUTRIA	Ñ  EL ÑANDU	comparte
 ESCORNABOT	O  EL OSO	P  EL PERRO	Q  EL QUOL TIGRE

tablero-
alfabeto-a-
i.svg

tablero-
alfabeto-a-
i.pdf

tablero-
alfabeto-j-
q.svg

tablero-
alfabeto-j-
q.pdf

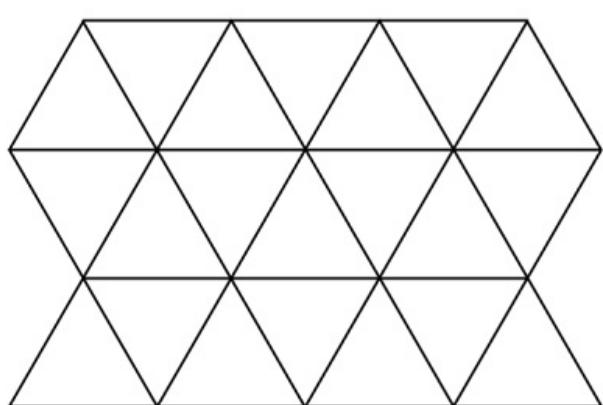
tablero-
alfabeto-r-
z.svg

tablero-
alfabeto-r-
z.pdf

	alfabeto-r-z.svg	alfabeto-r-z.pdf
--	----------------------------------	----------------------------------

Tablero triangular

Para usar la plantilla triangular debes cambiar la programación para que los grados en los giros se adapten al dibujo, el robot debe ir por las líneas y llegar a los vértices de los triángulos.

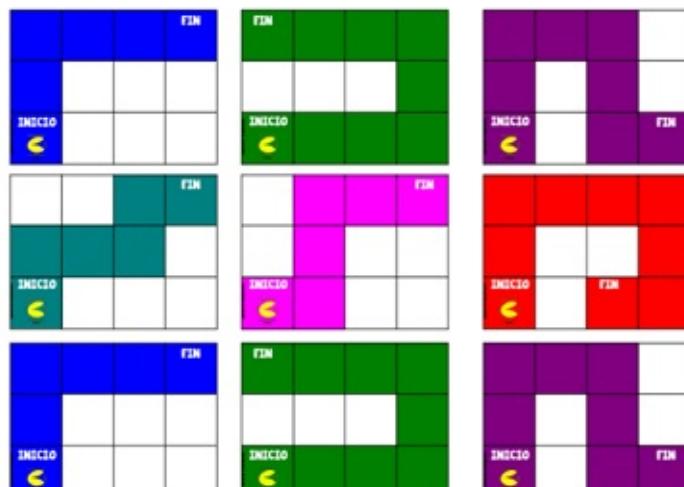
Tablero A-3	Archivo SVG	Archivo PDF
	Plantilla triangular formato .SVG	Plantilla triangular formato .PDF

Tablero con papel pintor

Hay que cortar en papel las cuadrículas por donde queremos que se mueva el robot, las pegaremos en papel de pintor y después pintarán un mapa por el que se desplazará Escornabot.



Tarjetas inicio-fin formato Word



Tarjetas recorridos formato .PDF



Tarjetas frutas formato Word



[Tarjetas ecosistemas formato .PDF](#)

Disfraces

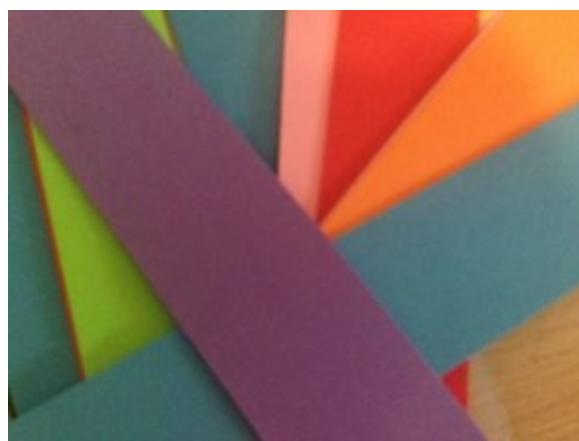
El origen de todo esto viene de [@mininacheshire](#) y del blog de Lobotic

Material necesario

- Goma eva
- Tijeras o cutter
- Regla
- Pegamento
- Folios
- Pinturas
- Ojos molones
- Cualquier objeto que resulte útil

Manos a la obra

Un A3 de goma eva mide 30cm de ancho y da de sobra para cubrir el robot de lado a lado. Los 42 cm de largo permite cortar 6 tiras de 7 cm en cada plancha. Yo hago los cortes con cúter, pero si van a trabajar los pequeos con tijeras mucho mejor



Disfrazando la versión DIY

Para simplificar usaremos las piezas para fijar el disfraz, en concreto la bola y la placa.



Las propias piezas nos van a servir de plantilla para cortar su silueta. Colocando la tira sobre el robot puedes calcular donde quieras que quede la tira, en mi caso los puse para que se quedara el corte en la parte de la bola:

Cableado

Haremos unos agujeros para pasar los cables de las baterías y los que conectan a la placa.



Buzzer o altavoz

El buzzer o zumbador lo podemos pinchar directamente en la plancha y utilizarlo de nariz. Por la parte trasera se conectan los cables sin problema quedando bastante sujeto.



Ojos

Yo no puedo evitar usar estos ojos pero cada uno debe utilizar su imaginación. Para las cejas he utilizado el mismo cable de arduino haciendo dos agujeros donde anclar los extremos, después puedes darle la forma que quieras.



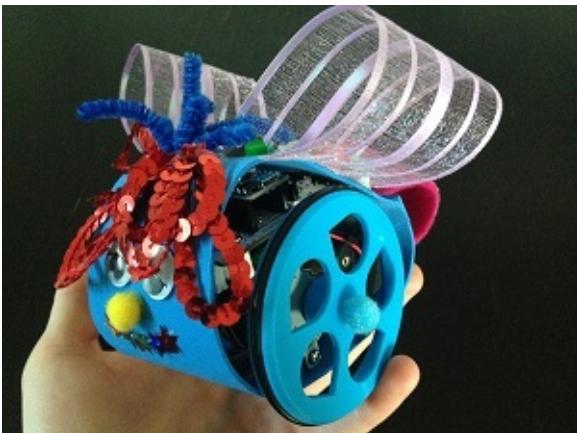
Boca

Buscando un poco por Internet tienes un montón de imágenes con bocas para imprimir, recortar y pegar.



Cuerpo

Utiliza trozos de goma eva para adornar el cuerpo, folios pintados o cualquier objeto que puedas pegar y quede molón.

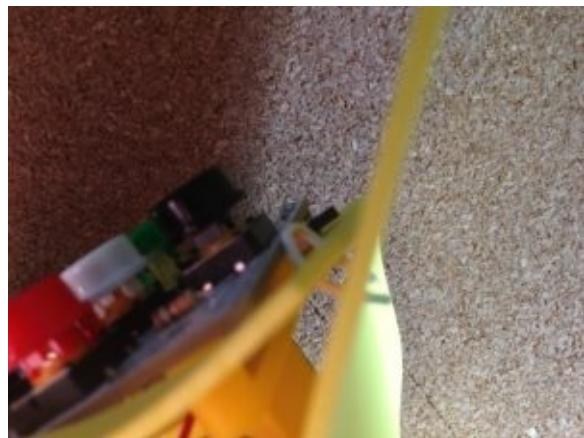


Disfrazando la versión 2.12

En este caso hay que tener en cuenta que la placa es rectangular y más grandota. Como en la anterior uso las propias piezas para anclar la goma eva.



Al no tener cables necesitas menos trozo de largo para cubrir el robot entero. Yo he utilizado ese sobrante para fabricarle una colita usando un cable como fijación para poder quitar y poner con facilidad.



Ya tienes una base sobre la que trabajar, ahora imaginación al poder.



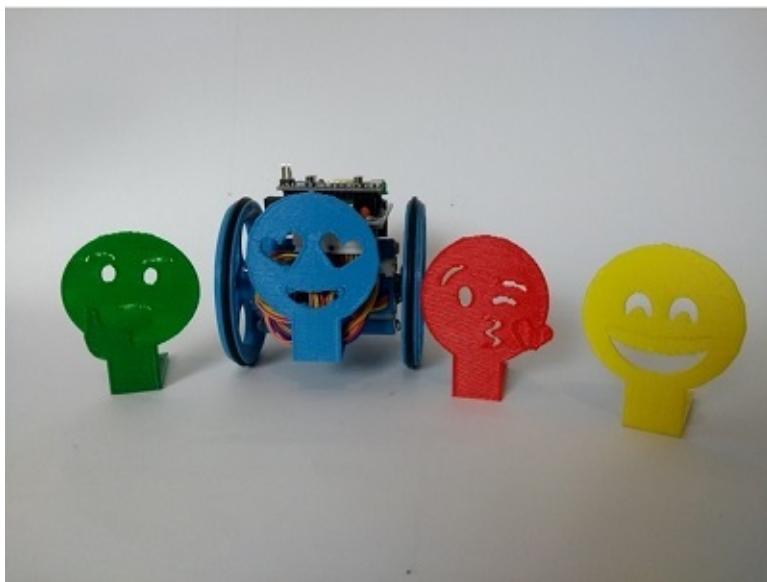
Otros ejemplos

En la wiki oficial tienes muchos ejemplos que te pueden dar ideas, te dejo enlace a [máscaras Escornabot](#)

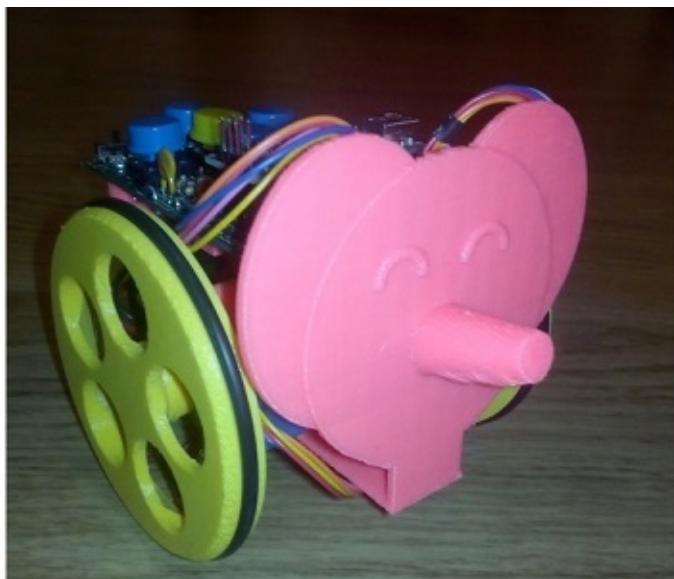
Máscaras Impresión 3D

Puedes consultar la colección completa de máscaras en la [cuenta Escornabot de Thingiverse](#)

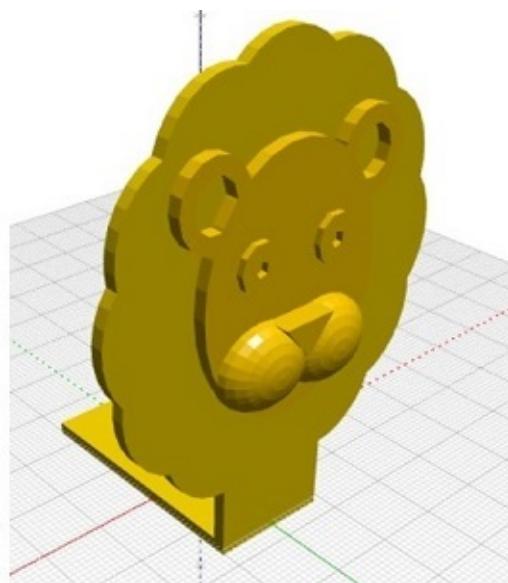
[Escornamoji por Programo Ergo Sum](#)



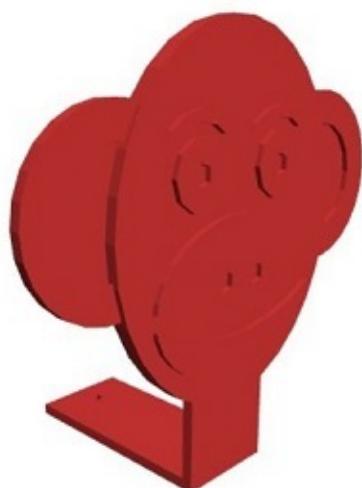
[Elefante por Lobotic](#)



[León por Lobotic](#)



[Mono por Lobotic](#)



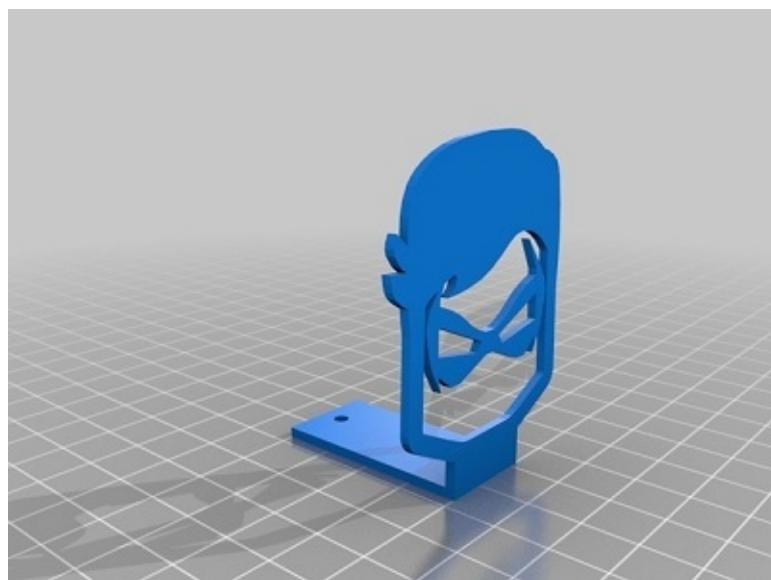
[Spiderman por Lobotic](#)



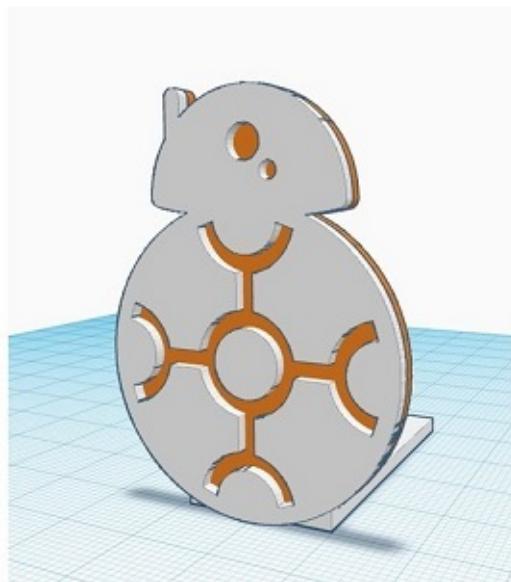
Batman por Angel



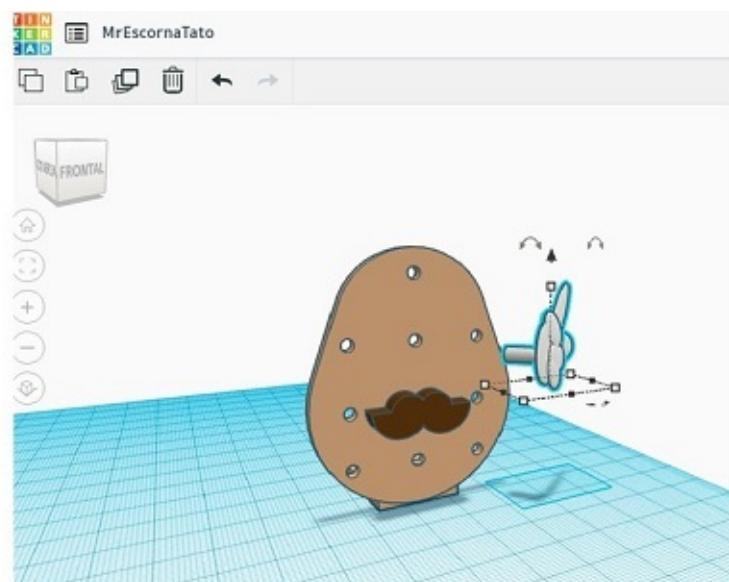
Robin por Angel



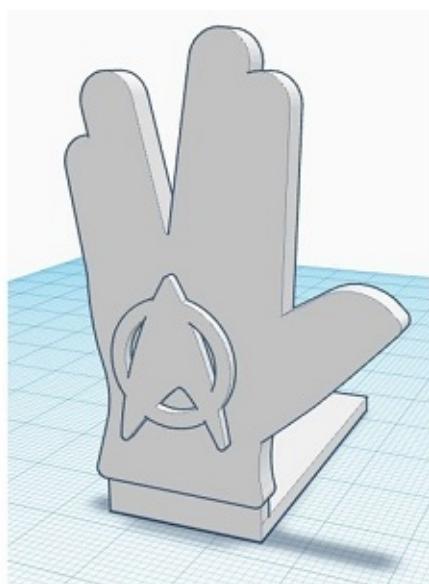
BB-8 por Angel



Mr Escorna Tato por Angel

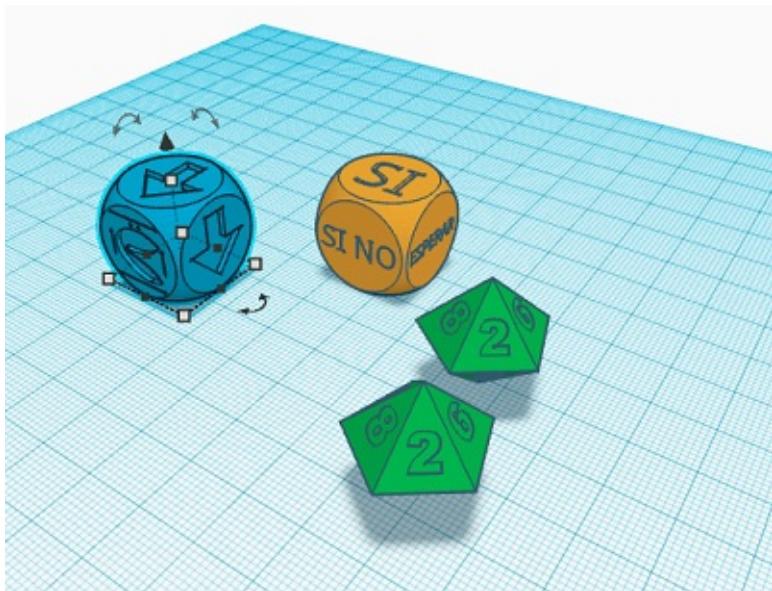


Treky por Angel



Dados CoDices

Angel ha compartido en su [GitHub CoDices](#), son unos dados modelados en 3D que puedes imprimir para trabajar el pensamiento computacional.



Diseño de ruedas

Lo bonito de este tipo de proyectos, es que se puede iniciar en diferentes campos como el diseño 3D.

Un programa gratuito y de fácil aprendizaje muy recomendable para iniciarse en el diseño 3D es [Tinkercad](#). Puedes consultar esta [guía de uso de Of3lia](#)

Si te atreves con algo más complejo, puedes aprender a usar Freecad con Obijuan [Tutorial Freecad I](#) y [Tutorial Freecad II](#)

De vez en cuando, **realizamos concursos como el Reto Escorna Rueda**. Para que cualquier persona diseñe su propia rueda y la comparta con la comunidad.

Si tienes una impresora 3D, descarga los archivos para imprimir en esta [colección de ruedas en la cuenta Thingiverse del proyecto](#).



Ruedas de [María Loureiro](#)

Añadir módulo Bluetooth BLE

Guía de referencia, con enlaces al Bluetooth necesario [aquí](#)

Material necesario

- 2 cables macho-macho de 10cm
- 4 cables macho-hembra de 10cm
- 1 **Bluetooth Ble HM-10 o compatible**
- APP Escornabot para [Android](#) o [IOS](#)

Importante - Esta aplicación funciona solamente con módulos BLUETOOTH BLE, si utilizas un Bluetooth normal como HC-05 o HC-06 [lee esta entrada de Lobotic](#)

2 cables macho-macho

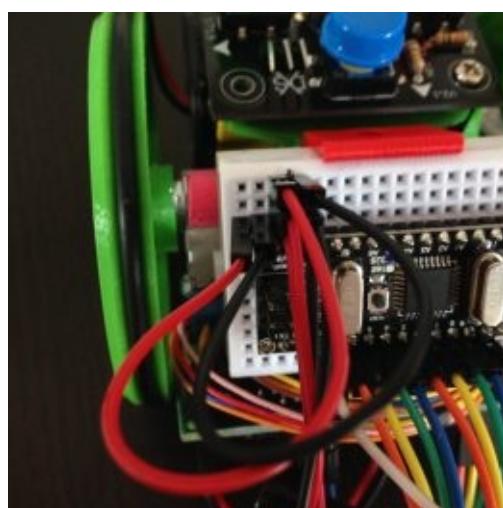
Primero necesitaremos los dos cables macho-macho (con pincho en los dos lados) para ampliar los GND disponibles y así poder usar uno para el Bluetooth.

Si tienes dudas sobre el funcionamiento de la protoboard [visita de nuevo su apartado en el curso](#).

Pincharemos el **cable rojo encima del rojo y el negro encima del negro** de las [placas de los drivers](#).

Puede que en el kit el color de los cables sea diferente. No importa, lo que cuenta es colocar cada cable en su posición. El único motivo de intentar utilizar cableado del mismo color es facilitar el montaje del robot.

Después en las dos filas que tenemos libres pincharemos los otros extremos de las puntas, de este modo tendremos dos filas libres en la parte superior:



- Para conectar el cable del portapilas.
- Para conectar el GND del Bluetooth.

4 Cables macho-hembra de 10cm y 1 módulo Bluetooth BLE HM-10 o compatible

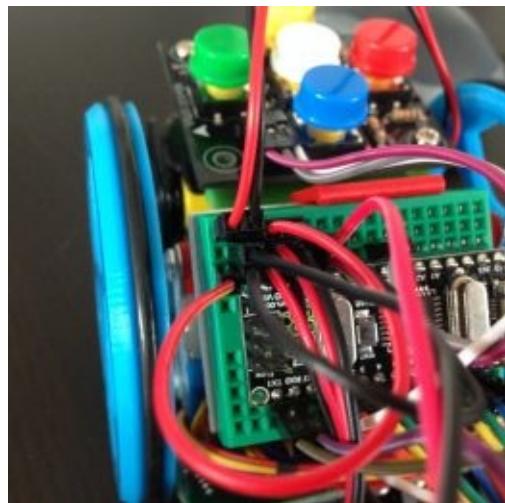
Relación de conexiones entre las patillas del Bluetooth y el Arduino Nano:

Bluetooth BLE	Arduino NANO
RX	TX
TX	RX
GND	GND
VCC	5V

Utilizaremos los **cables de color negro** para GND o negativo y los **de color rojo** para VIN, VCC, 5V o positivo.

El **cable negro** lo pinchamos en la **patilla GND del Bluetooth y la otra parte encima del negro** que hemos ampliado anteriormente.

El **cable rojo** lo pinchamos en la **patilla VCC del Bluetooth y la otra parte encima del cable de corriente 5V** del teclado.



Usaremos otros dos cables macho-hembra para conectar las patillas RX, TX del Bluetooth con las del Arduino NANO.

- La **patilla RX del Bluetooth** la conectamos **al TX del Arduino**.
- La **patilla TX del Bluetooth** la conectamos **al RX del Arduino**.

Cambios y verificaciones en la programación

Entramos en la pestaña "**Configuration.h**"

Buscamos la parte del código que muestra la imagen:



```
////////// general configuration
////////// engine to use
#define ENGINE_TYPE_STEPPERS

// button set to use (analog input, digital input)
#define BUTTONS_ANALOG
//#define BUTTONS_DIGITAL

// milliseconds after a button is considered as pressed
#define BUTTON_MIN_PRESSED 30
```

Dejamos la línea `#define BUTTONS_ANALOG` para poder utilizar también el teclado

Revisamos si tenemos activada la opción Bluetooth y los baudios a los que enlazará

En la mayoría de los casos funcionan a 9600 baudios que es el valor por defecto que lleva la programación.

Si quieras saber como comprobar los baudios a los que esta programado vuestro módulo [visita esta entrada de Makespace Madrid](#)



```
// point of view set when Vacalourabot is started
#define POV_INITIAL    POV_ESCORNABOT

// bluetooth serial
#define USE_BLUETOOTH true
#define BLUETOOTH_BAUDS 9600
```

Si en la pestaña "Configuration.h" no vemos las líneas que aparecen en la imagen las debemos añadir.

Activar línea Serial

Dentro de la pestaña "Configuration.h", al final buscamos este apartado:



```
///////////////////////////////
////// Button set Bluetooth
///////////////////////////////

#ifndef USE_BLUETOOTH

// Arduino serial port (default: Serial)
#define BLUETOOTH_SERIAL Serial
//#define BLUETOOTH_SERIAL Serial1
//#define BLUETOOTH_SERIAL Serial2
//#define BLUETOOTH_SERIAL Serial3

#endif // USE_BLUETOOTH
```

Quitamos las dos barras inclinadas de la línea:

```
//#define BLUETOOTH_SERIAL Serial
```

Para saber si queda activada, la línea pasa de color gris a negro y la palabra Serial final se pone en amarillo.

Instalación aplicación Escornabot

Descargamos la aplicación gratuita en nuestro móvil o tablet:

- [APP Escornabot para Android](#)
- [APP Escornabot para iOS](#)

Conecrtar alimentación y vinculación Bluetooth con APP

Ya tenemos todo preparado, lo siguiente:

- **Conecrtamos la alimentación del robot**, justo encima de la ampliación que hicimos al inicio. El robot debería dar el pitido del check inicial y el led rojo del Bluetooth comenzar a parpadear.
- Entramos en la aplicación, buscamos el dispositivo Bluetooth y nos conectamos, una vez enlazados **la luz del Bluetooth dejará de parpadear para quedarse fija**.



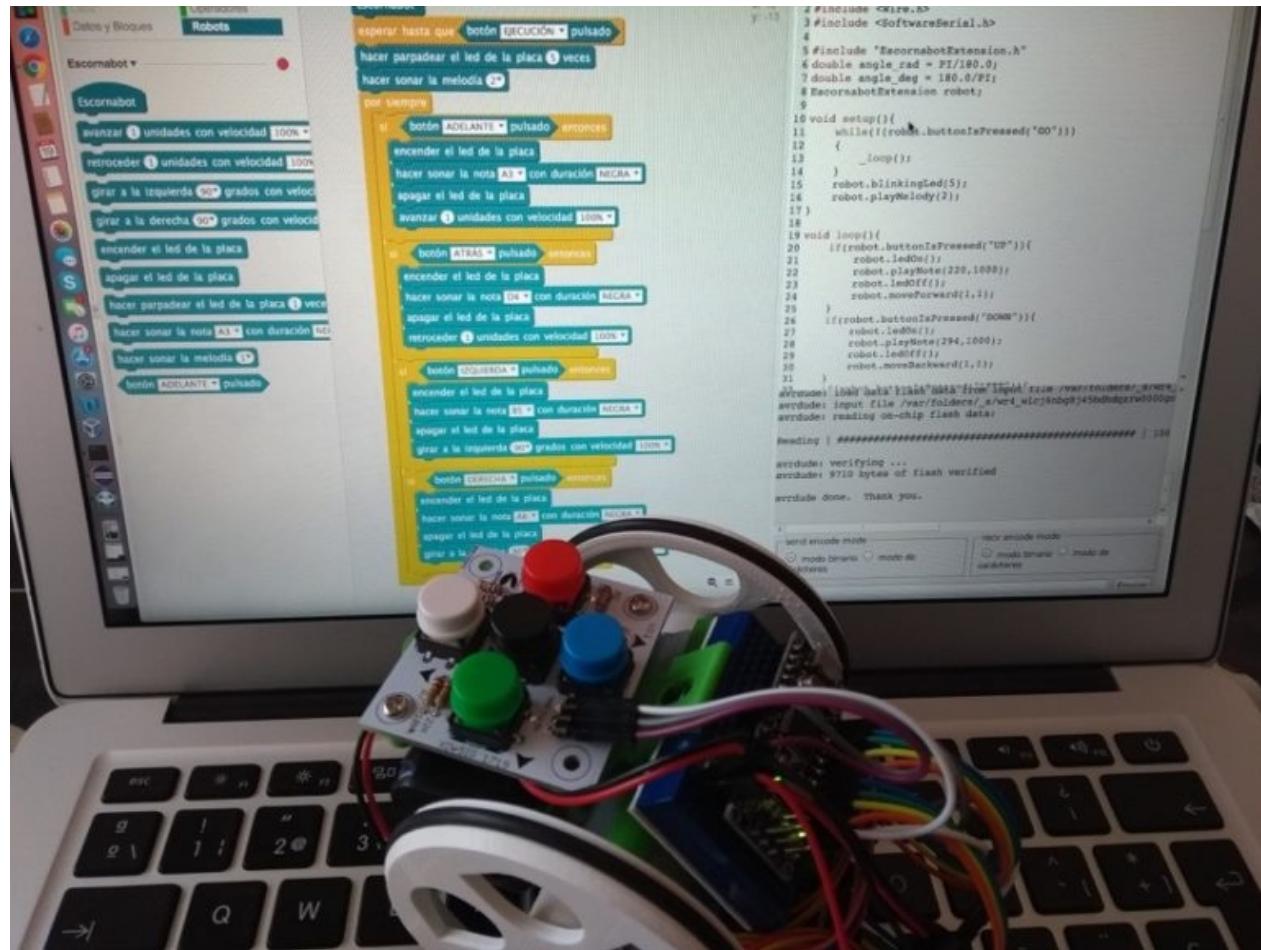
- Pulsamos en la parte inferior, en el dibujo del círculo con las flechas y pasamos a la pestaña del teclado. Si todo ha ido bien cuando pulsemos cada tecla en nuestro teléfono el robot emitirá un pitido y al pulsar el botón blanco comenzará a caminar.



Ya tenemos nuestro Escornabot con conexión Bluetooth.

Programación con mBlock

Las personas que vayan más avanzadas y necesiten un extra, pueden probar a programar Escornabot con bloques a través de [mBlock](#).



Puedes consultar la [guía completa mBlock para escornabot](#) realizada por [Angel Villanueva](#)

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](#), 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 ESCORNABOT es LA ALTERNATIVA LIBRE a robots tipo Beebot, Colby pero también permite su programación con mBlock y código por lo tanto se puede usar en Infantil programando movimientos con los botones igual que Beebot y Colby o se puede programar utilizando mBlock y código utilizando este robot a niveles superiores, por lo tanto tenemos un robot **muy flexible** para usarlo en las aulas:

RoboTICa

Oferta de formación en Pensamiento computacional del Centro Aragonés de Tecnologías para la Educación.



Créditos

Autoría

- Pablo Rubio Martínez

Colaboradores:

Cualquier observación o detección de error por favor aquí soporte.catedu.es

Los contenidos se distribuyen bajo licencia Creative Commons tipo BY-NC-SA.



**GOBIERNO
DE ARAGÓN**

Departamento de Educación,
Cultura y Deporte

