Programando DuinoBot v2.4 con mBlock

Ing. Pouso Marcelo A.
mapouso86@gmail.com

Versión 0.7

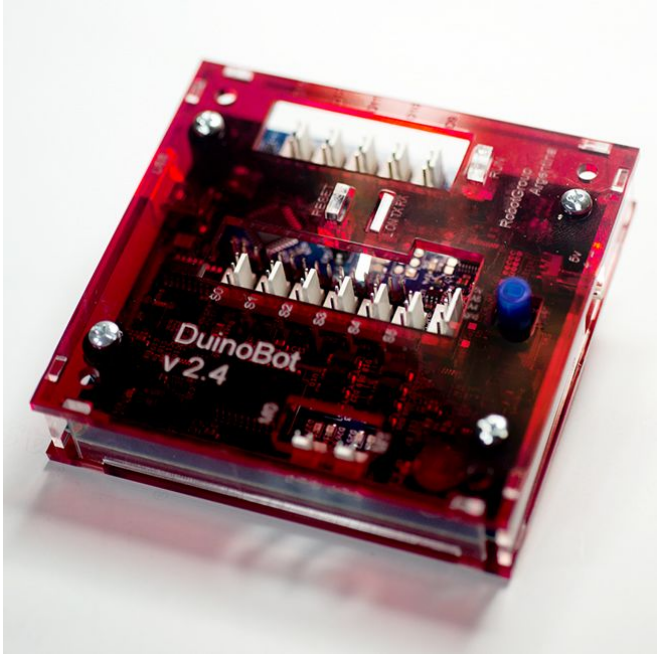
Introducción

En el presente documento se detallan los pasos a seguir para programar la placa DuinoBot v2.4 de la empresa RobotGroup[1] con el software mBlock[2] de la empresa Makeblock[3].

Para tal propósito se creó un extensión que agrega bloques de programación específicos para la placa DuinoBot. Dicha extensión se puede descargar desde el centro de extensiones de mBlock o pedir por mail a mapouso86@gmail.com. (Ver Integrando DuinoBot a mBlock)

DuinoBot 2.4

La placa DuinoBot está basada en Arduino Nano[4] y posee un microcontrolador ATmega328. Además, trae integrado un controlador de motores L298, 1 pulsador de propósito general, 5 pines digitales, 6 entradas analógicas, 1 led de propósito general y dos salidas para motores de CC.

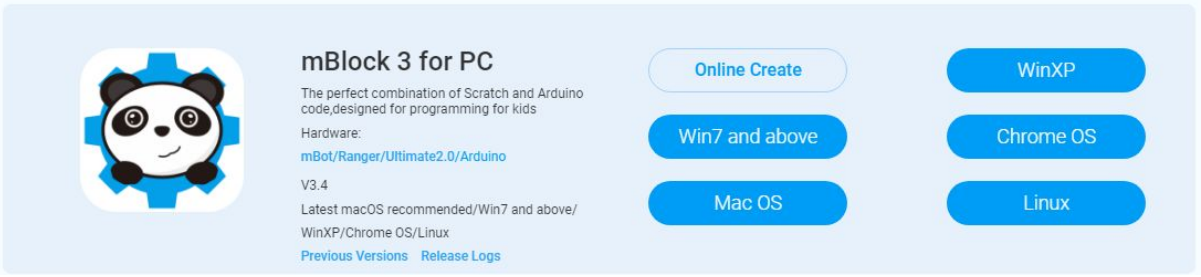


mBlock v3.4

[1] https://robotgroup.com.ar
[2] http://www.mblock.cc
[3] https://www.makeblock.com/
[4] https://store.arduino.cc/usa/arduino-nano

mBlock es un entorno de programación basado en Scratch[5] 2.0 que se puede descargar gratuitamente desde la web www.mblock.cc

La versión utilizada para programar es "*mBlock 3 for PC"* descargada desde http://www.mblock.cc/mblock-software/



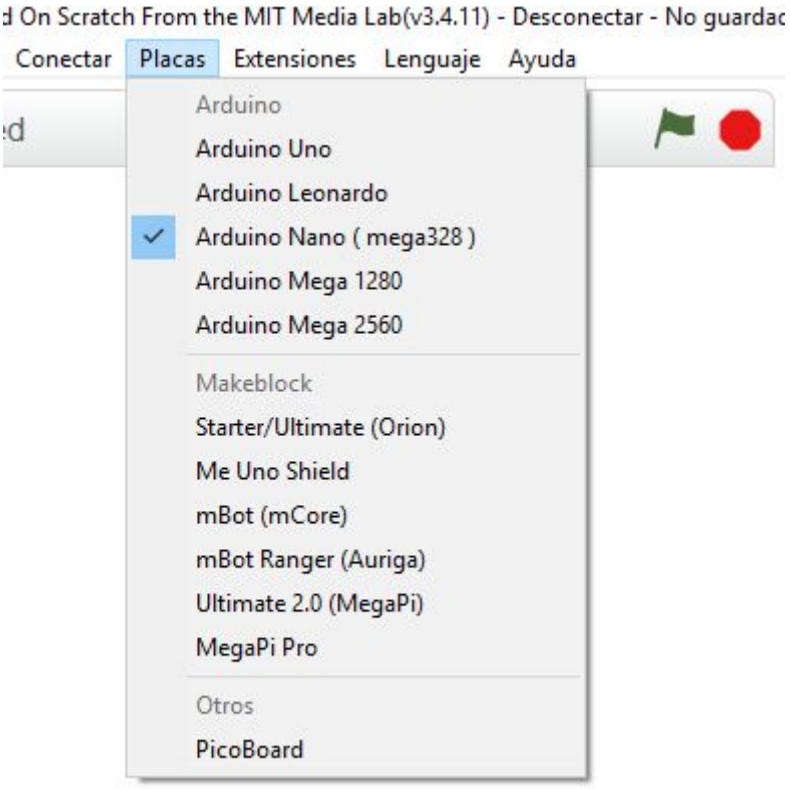
El programa se encarga de instalar todas las librerías y drivers de Arduino.

Integrando DuinoBot a mBlock

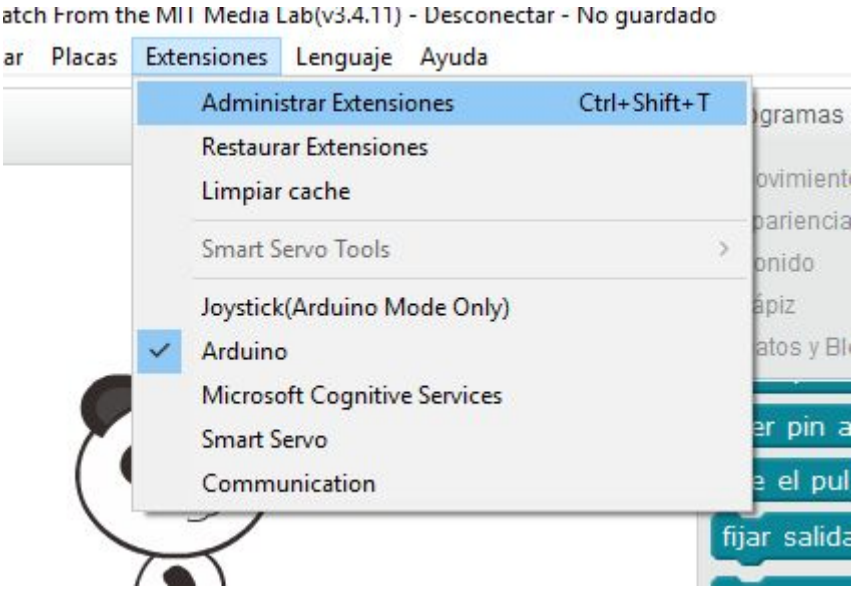
Una vez descargado e instalado mBlock v3.4 podemos agregar la extensión desde internet o localmente con el archivo zip (*DuinoBot_v2_4.zip*), para esto ejecutamos mBlock y seguimos los siguientes pasos:

Primero debemos configurar la placa como **Arduino Nano** desde el menu *Placas → Arduino Nano (mega328)*

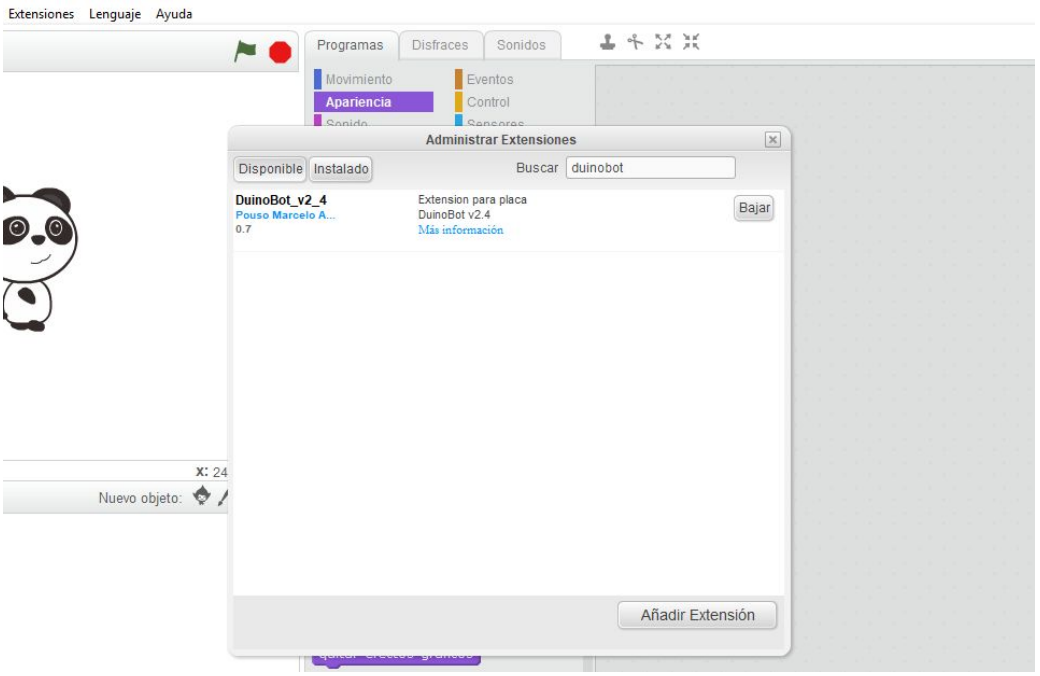
[5] https://scratch.mit.edu/



Luego, debemos descargar/agregar la extensión **DuinoBot**:
Nos dirigimos a *Extensiones → Administrar Extensiones*

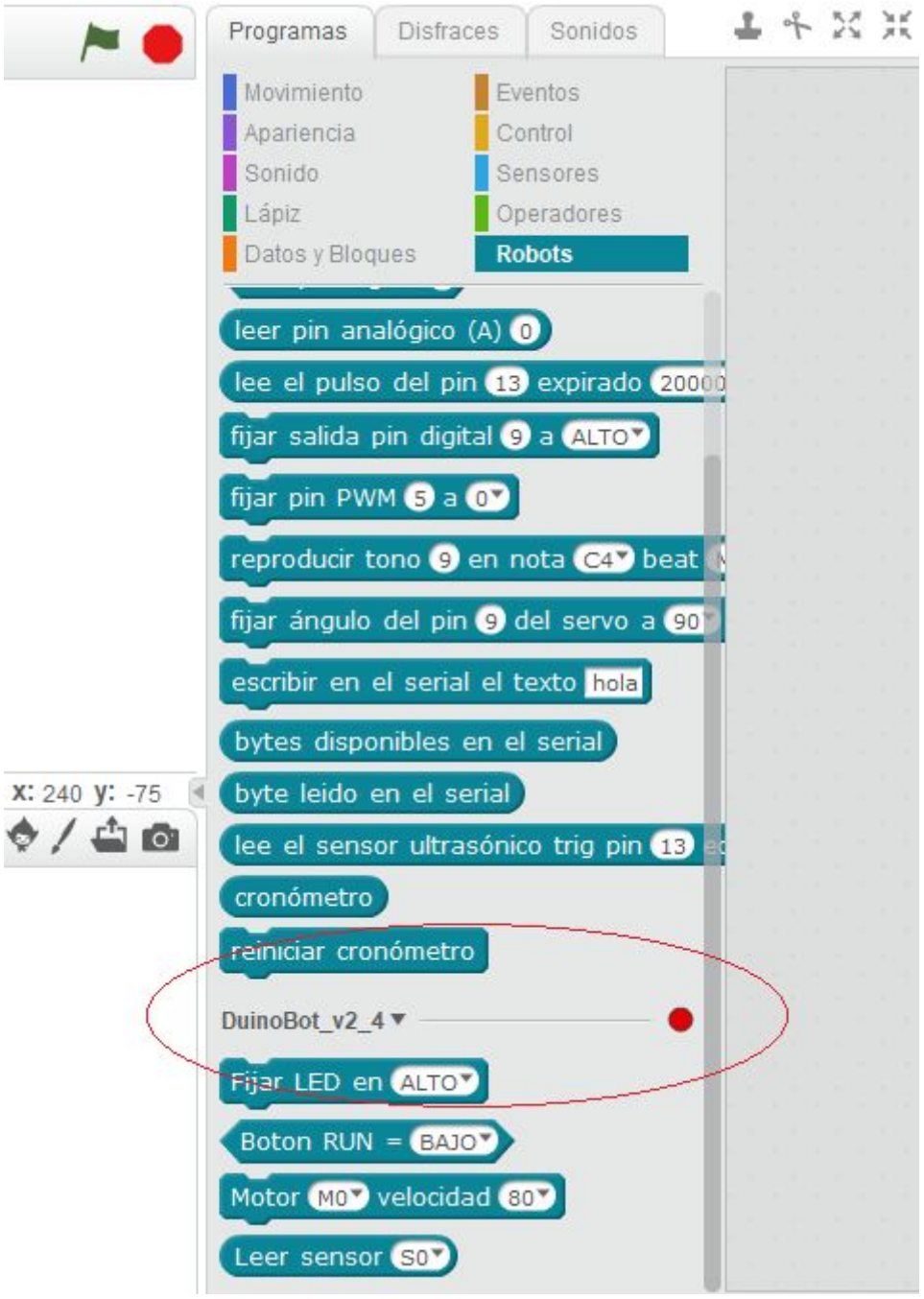


Podemos descargar la extensión directamente desde el centro de extensiones de mBlock (para lo cual necesitamos estar conectado a internet) o podemos buscar el archivo *DuinoBot_v2_4.zip* en nuestra computadora haciendo click en *“Añadir Extensión”.*



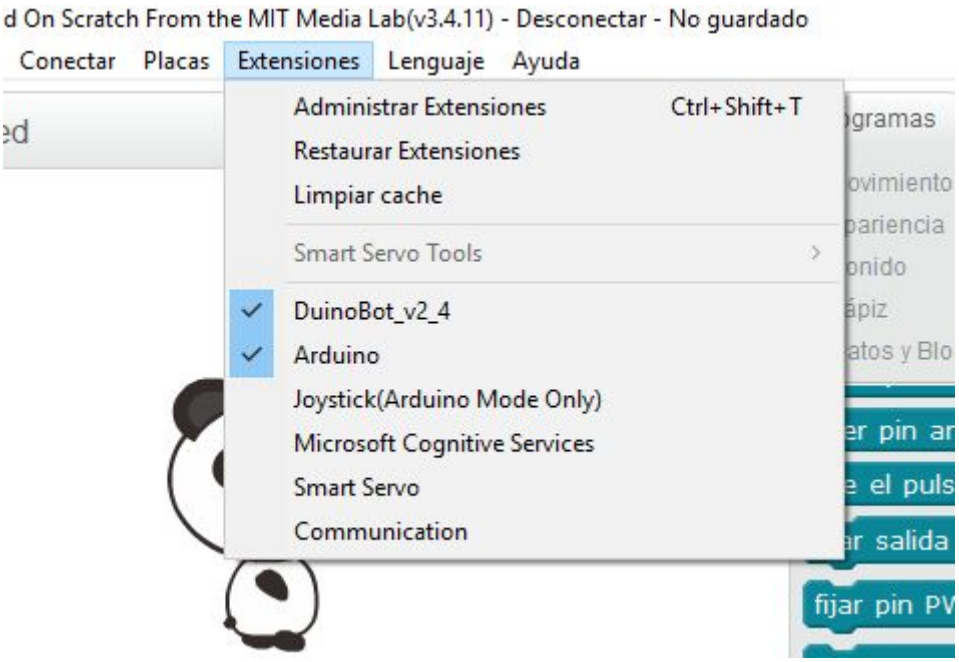
Si poseemos conectividad a internet colocamos *"duinobot"* en la entrada *Buscar* y nos debería aparecer la extensión **DuinoBot_v2_4**, damos click en botón *Bajar*, luego cerramos el cuadro *"Administrar Extensiones"*.

Una vez finalizado, en el menú programación solapa *Programas* → *Robots* aparecen los bloques de control de nuestra placa.

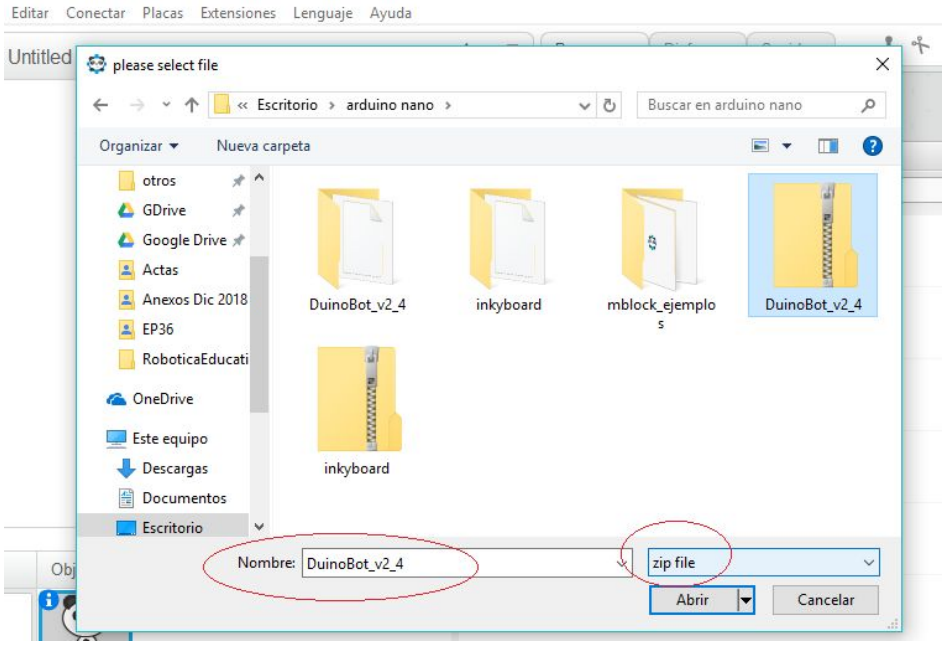


Como observamos la extensión nos permite controlar el led de la placa y pulsador (botón) de propósitos generales, como también leer los sensores y controlar los motores. También podemos utilizar todos los bloques Arduino (“fijar pin PWM”, “escribir en el serial..”, etc)

Esta extensión se puede habilitar y deshabilitar desde el menú Extensiones:

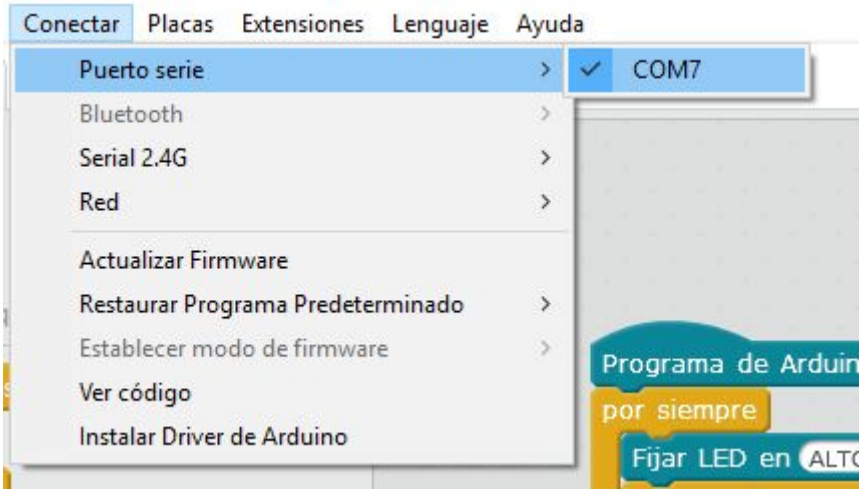


Si no se posee conectividad a internet para descargar la extensión, la misma se puede instalar desde un archivo zip (enviar mail para recibir carpeta zip)

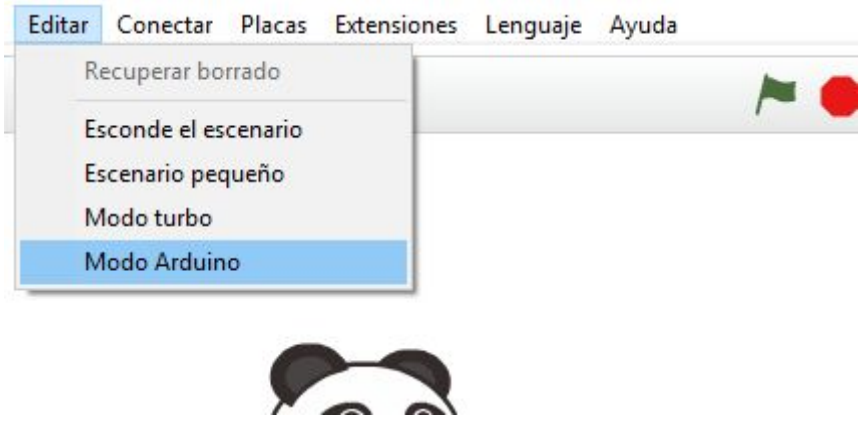


Programando la placa

Con la placa conectada al puerto USB debemos conectarnos con mBlock desde el menú *Conectar → Puerto serie...* y seleccionamos el puerto al cual está conectada placa.

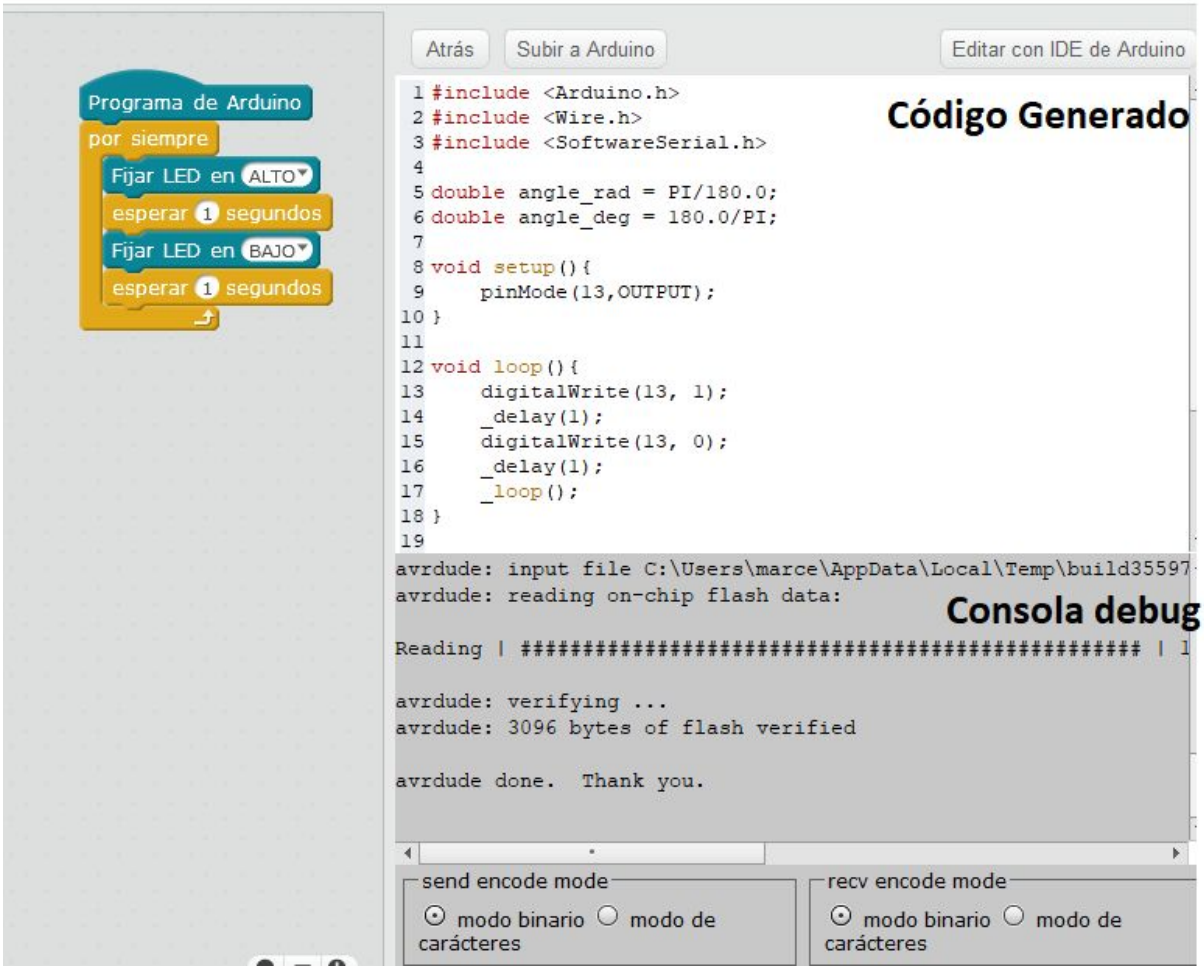


Una vez realizado nuestro programa con los bloques de programación que nos proporciona la interfaz debemos subir la programación a la placa, para ello podemos hacer doble click sobre el bloque *"Programa de Arduino"* o ir al menú *Editar → Modo Arduino*



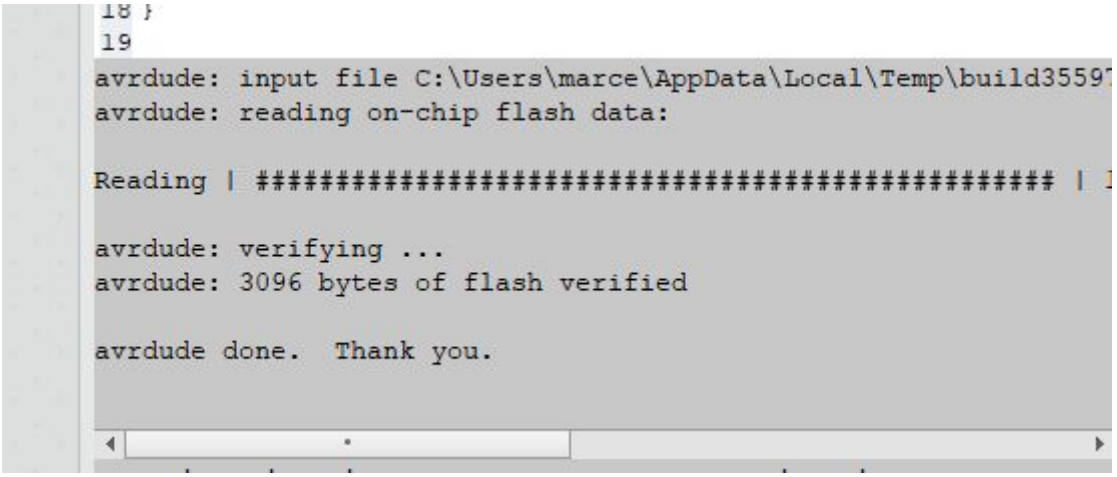
.

El *Modo Arduino* nos permite ver en la ventada de la derecha el código generado a partir de nuestros bloques y una consola de debug.



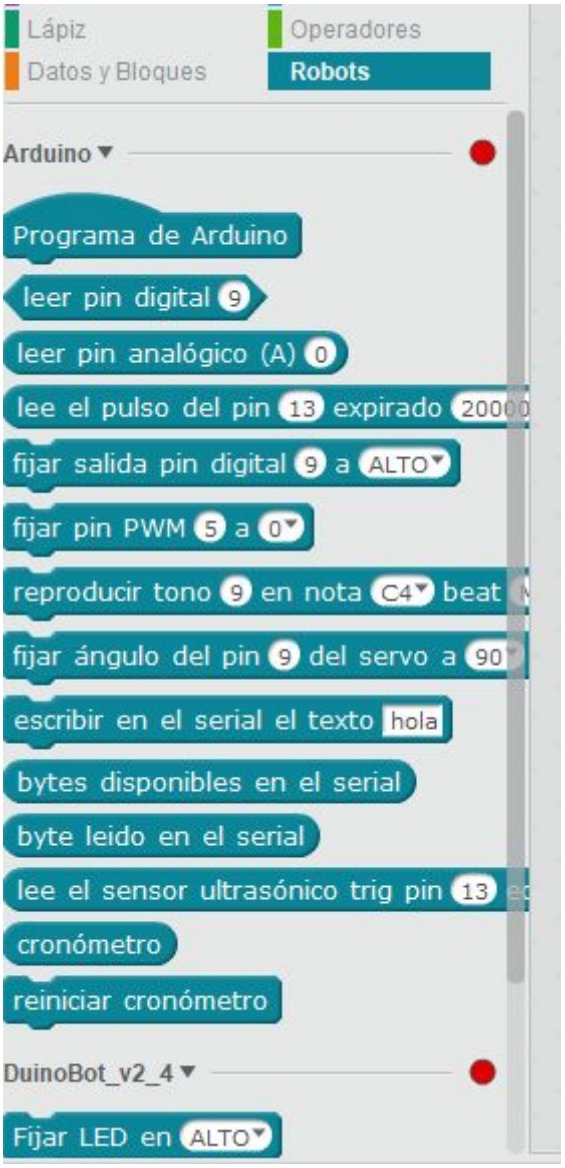
Desde este modo podemos grabar nuestro programa con el botón ***"Subir a Arduino"***. Nos aparecerá una ventana que dice al principio "Subiendo" mientras está subiendo en la placa y finalmente dice "Programación finalizada" cuando termina de subir el programa a la placa.

Si la programación fue correcta deberíamos ver en la consola el mensaje *"Thank you"*, de otra manera aparecerá un mensaje de error.



Programación

La programación se realiza de la misma manera que con Scratch 2.0 pero teniendo a nuestra disposición los bloques correspondiente al ítem **“Robots”**, en la cual vemos los bloques propios de **Arduino** y debajo los de **DuinoBot**. Ambos conjuntos sirven para programar la placa DuinoBot.



mBlock permite dos modos de uso: modo Scratch y modo Arduino.

El **modo Scratch** permite que nuestro personaje o historia interactúe con la placa, previamente se debe instalar un firmware en la placa y la misma debe estar siempre conectada mientras corremos el programa. La programación comienza con el bloque "***Al presionar...***". Este documento no abarca la explicación de este modo.

El **modo Arduino** sirve para programar la placa quedando el programa grabado en la placa. Esta programación siempre debe comenzar con el bloque ***"Programa de Arduino***"

Al trabajar en modo Arduino es recomendable ir a "Editar" → "Modo Arduino" para tener un entorno de trabajo más amplio.

Control LED

En siguiente ejemplo se observa la programación para hacer titilar el led de propósito general en la placa cada 1 segundo.

Programando pulsador

La placa incorpora un pulsador (botón) denominado RUN que originalmente se utiliza para comenzar ejecutar la programación realizada con miniBloq [6].

En el caso de la programación con mBlock dicho pulsado puede ser de uso general.

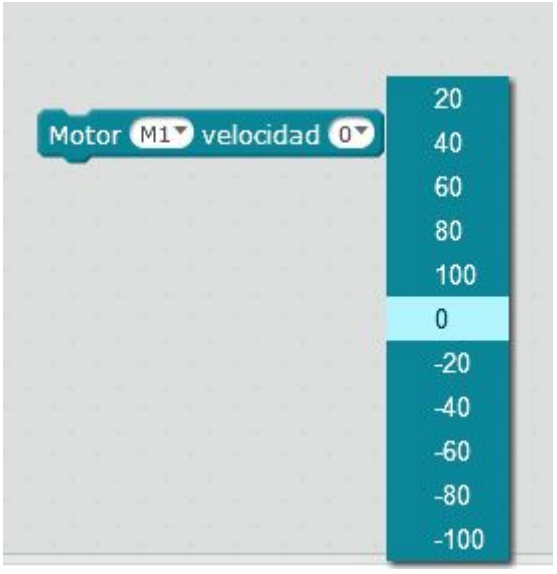
En el siguiente ejemplo el pulsado es utilizado como en su uso original, ejecutar la programación luego de ser pulsado:

[6] http://blog.minibloq.org/

En el siguiente ejemplo, el led se enciende cuando el pulsador es pulsado.

Controlar Motores

Podemos controlar los dos motores de CC, configurando su velocidad normalizada entre 0 y 100, siendo valores negativos para cambiar el sentido de giro de los mismo.



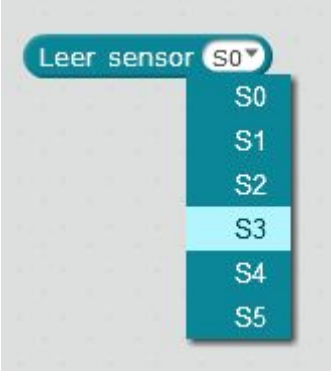
En el siguiente ejemplo, se espera a que se pulse el botón RUN y por siempre se ejecutará la siguiente secuencia:

- Motor 0 a una velocidad 100 (máxima) giro positivo y Motor 1 a una velocidad 60 giro negativo durante 5 segundos.
- Motor 0 a una velocidad 100 (máxima) giro negativo y Motor 1 a una velocidad 80 giro positivo durante 10 segundos.
- Se configuran los dos motores a velocidad 0 (parados) durante 15 segundos.

Lectura de Sensores

La placa DuinoBot nos permite conectar 6 sensores a las entradas S0...S5.

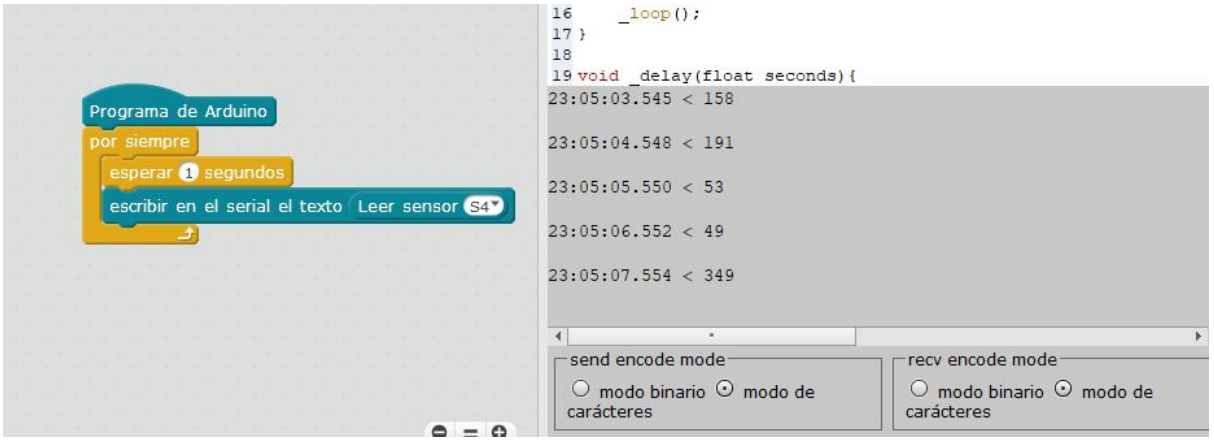
Mediante el bloque *"Leer sensor SX*" podemos obtener los valores de esas entradas.



En el siguiente ejemplo se imprimen en la consola serie los valores leídos del sensor conectado a la entrada S4.

Para este ejemplo se utiliza también el bloque *“escribir en serial el texto ...”* propio de Arduino.

Una vez programada la placa de la debe re-conectar al puerto serie (menú *Conectar → Puerto serie*) y configurar en la consola con la opción: *recv encode mode: modo de caracteres.*



Programando Pines Digitales

En la placa DuinoBot aparecen como pines digitales accesibles los pines correspondientes a D9, D10, D11, D12 y D13. Los mismo podemos controlarlos usando los bloques Arduino. Ejemplo: hacer titilar el módulo led conectado al conector D9.

Programando Pines PWM

Los pines D9, D10 y D11 pueden ser programado con PWM.

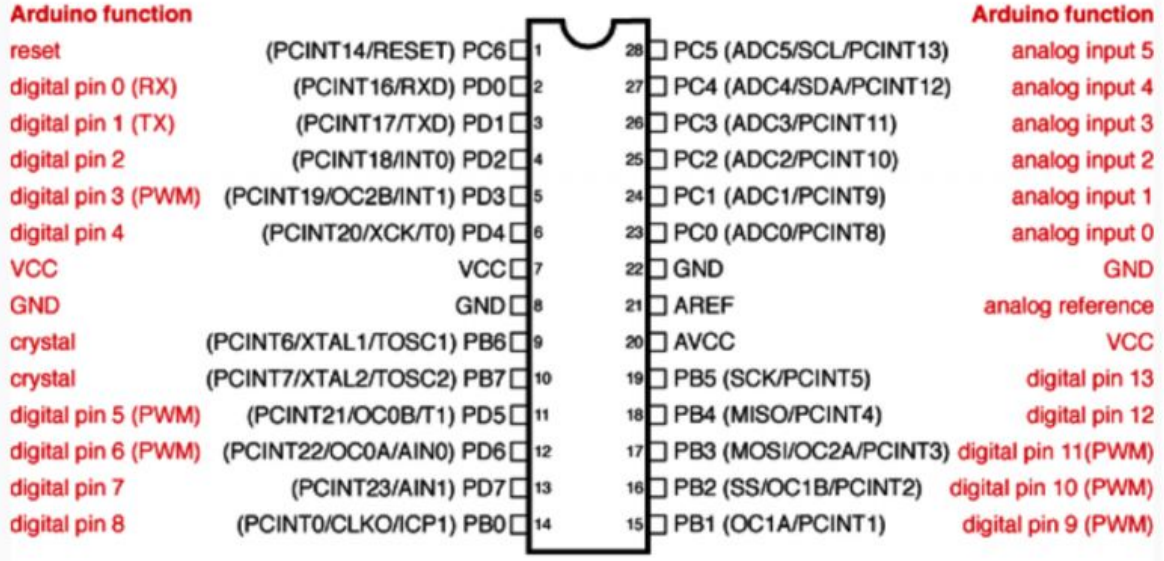
En el siguiente ejemplo se incrementa el valor de PWM (D10) cada vez que se presiona el botón RUN. Si conectamos la módulo led podemos observar como aumenta el brillo del led cada vez que presionamos el botón RUN.

Anexo

La Placa DuinBot v2.4 se basa en Arduino Nano (mega328). En la siguiente tabla se detalla el pinout correspondiente.

Arduino Nano	DuinoBot
PC0 - ADC0	S0
PC1 - ADC1	S1
PC2 - ADC2	S2
PC3 - ADC3	S3
PC4 - ADC4	S4
PC5 - ADC5	S5
PB1 - D9 - PWM	D9
PB2 - D10 - PWM	D10
PB3 - D11 - PWM	D11
PB4 - D12	D12
PB5 - D13	D13*
PB5 - D13	LED PLACA (*comparte con D13)
PB0 - D8	BOTÓN RUN
PD6 - PWM	MOTOR 0 - ENABLE
PD5 - PWM	MOTOR 0 - D0
PD7	MOTOR 0 - D1
PD3 - PWM	MOTOR 1 - ENABLE
PD4	MOTOR 1 - D0
PD2	MOTOR 1 - D1

ATMEGA328P-PU Chip to Arduino Pin Mapping



Digital Pins 11,12 & 13 are used by the ICSP header for MISO, MOSI, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.