

# Empezando

Entramos en Proyectos y podemos ver nuestros proyectos creados como también empezar uno.



Y nos aparece tres opciones :



En esta ventana podremos elegir que tipo de proyecto vamos a realizar:

- **Proyecto Personal:** Iniciar un nuevo proyecto que sólo será accesible para el usuario. Posteriormente se puede compartir al resto de la comunidad si se desea.
- **Proyecto Profesor:** Iniciar un proyecto como profesor. De esta forma no se inicia un proyecto como tal, sino que se especifican los datos del proyecto y se genera un código para que los alumnos se puedan suscribir al proyecto. El profesor podrá supervisar y valorar los proyectos de sus alumnos.
- **Alumno:** De esta forma nos unimos a un proyecto planteado por el profesor. Nosotros realizaremos el proyecto como si de un proyecto personal se tratara, pero

el profesor podrá supervisar y valorar nuestro trabajo.

*Adaptado de [este enlace](#). José Andrés Echevarría @cantabRobots CC-BY-NC-SA*

Lo primero que tenemos que elegir es para qué tipo de placa se hace el proyecto.

- En el caso de que estés con el kit de CATEDU **Rover marciano con Arduinoblocks** el tipo de proyecto es para **ESP8266 / NodeMCU**
- En el caso de que estés con el kit de CATEDU **Arduino con Arduinoblocks** el tipo de proyecto es para **Arduino UNO**
- En el caso de que estés con el kit de CATEDU **ArduinoBlocks en el aula** tienes dos opciones totalmente válidas:
  - **ArduinoUno**
  - **ArduinoUno + Imagina TdR STEAM**
- En el caso de que estés con el kit de CATEDU **ESP32 en el aula** tienes dos opciones totalmente válidas:
  - **ESP32 STEAMakers**
  - **ESP32 STEAMakers + Imagina TdR STEAM**

## Nuevo proyecto personal

Tipo de proyecto	<input type="text"/>
Nombre	<div>Arduino Uno</div> <div>Arduino Nano / ATmega328</div> <div>Arduino Nano / ATmega328 (new bootloader)</div> <div>Arduino Mega / 2560</div> <div>Arduino Leonardo</div> <div>Arduino UNO + Imagina TdR STEAM</div> <div>3dBot / Imagina-Arduino</div> <div>Keyestudio EasyPlug</div> <div>Keyestudio KeyBot</div> <div>Otto DIY / Nano</div> <div>Otto DIY / Nano (new bootloader)</div> <div>ESP8266 / NodeMCU v2</div> <div>ESP8266 / WeMos D1</div> <div>ESP32 / WROOM</div> <div>ESP32 STEAMakers</div> <div>ESP32 STEAMakers + Imagina TdR STEAM</div> <div>ESP32 STEAMakers + 3dBot</div>
Descripción	
Componentes	

**ATENCIÓN** luego **NO** se puede cambiar. Es decir, un proyecto realizado para un tipo de placa, no se puede cambiar a otro tipo de placa (la razón es simple: las instrucciones cambian)

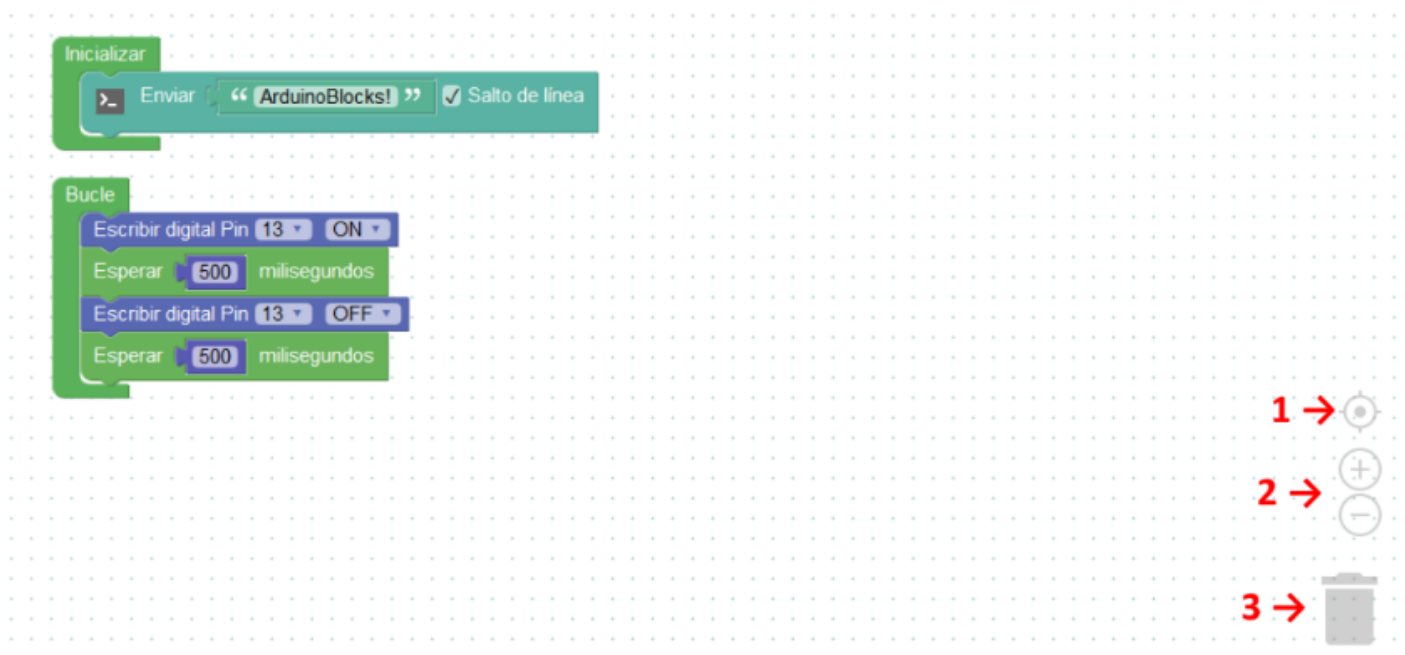
Luego el nombre y el resto de campos es optativo pero **importante y buena costumbre** rellenarlos, sobre todo si el proyecto lo **compartimos**:

- Descripción
- Componentes
- Comentarios

## Área de programación del proyecto

Este es el área sobre el que se trabaja en Arduinoblocks. En esta área arrastraremos y colocaremos los bloques que vamos a utilizar para crear nuestro programa.

En el área de trabajo hay un Zoom (2) para ampliar o reducir la imagen, un icono para centrar (1) y un icono donde podremos borrar los bloques que no utilicemos (3).



Adaptado de [este enlace](#). José Andrés Echevarría @cantabRobots CC-BY-NC-SA

Las principales secciones del área de programación son las siguientes :

HerramientasÁrea de programa

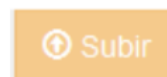
Bloque de iniciación



Bloque de bucle del programa principal

Opciones

Subir el programa a la placa Arduino conectada:



Puerto de conexión de la placa Arduino:



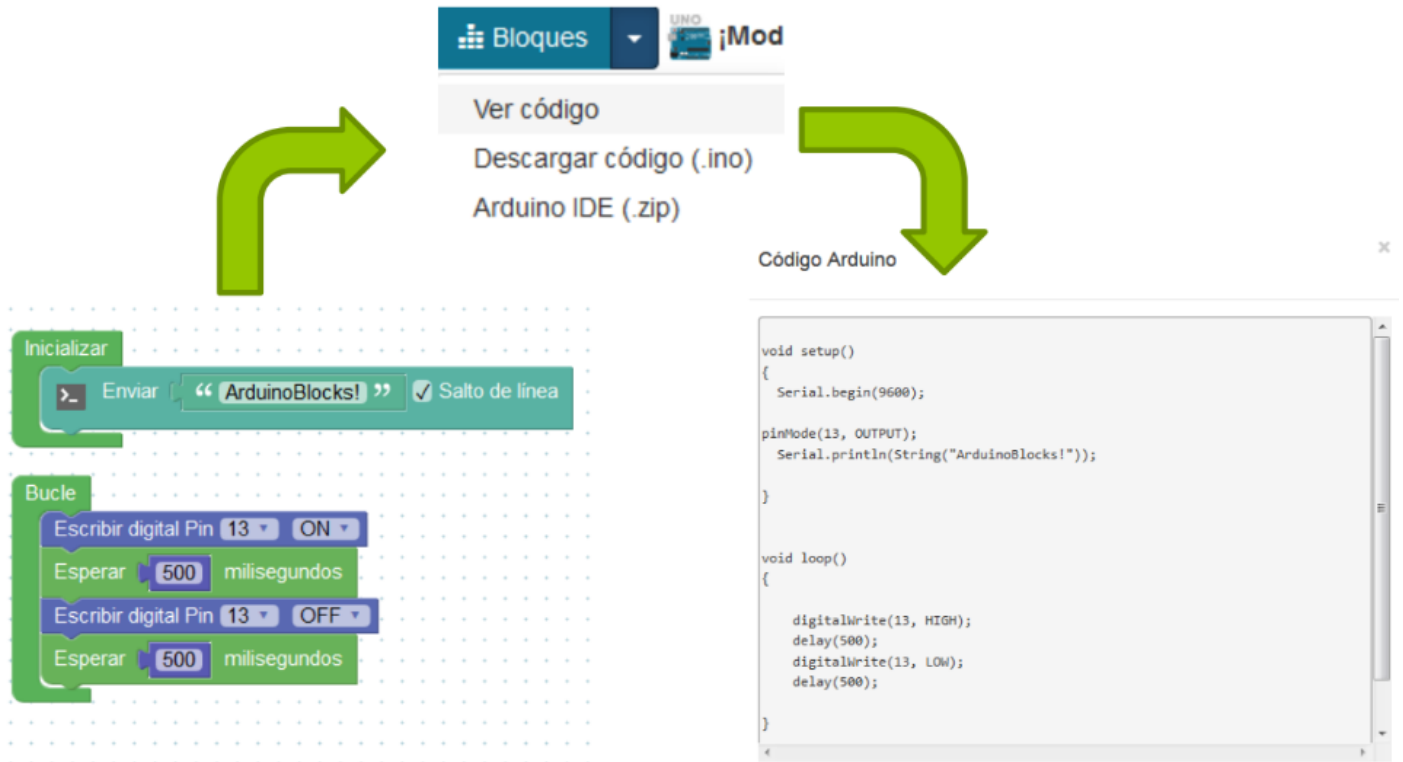
Mostrar la consola serie:



Adaptado de [este enlace](#). José Andrés Echevarría @cantabRobots CC-BY-NC-SA

## Ver el código

**ArduinoBlocks** genera el código de Arduino a partir de los bloques. El programa se puede compilar y subir directamente a la placa Arduino gracias a la aplicación **ArduinoBlocks-Connector**, sin embargo si deseamos ver o descargar el código podemos realizarlo desde el área de bloques.



Adaptado de [este enlace](#). José Andrés Echevarría @cantabRobots CC-BY-NC-SA

Siempre, desde un **lenguaje de programación en bloques** podemos obtener su equivalente a **Código de Arduino IDE** (de hecho es lo que hacen los programas), y luego con las funciones de **Código de Arduino IDE** el software lo pasa a **lenguaje máquina** que es la que se graba el Arduino, **pero no al revés** es decir, no existen programas que dado un código máquina o código Arduino IDE lo pasen a bloques gráficos, (igual que no hay programas que lean el código máquina que hay grabado en un Arduino y lo pasen a código Arduino IDE). Esto no es del todo 100% verdadero pues la Ingeniería inversa en informática trata pues de eso: obtener la fuente aunque sea parcial, pues si obtienes el código legible, puedes alterar lo que quieras.

Cuando compras un programa comercial, te dan el lenguaje máquina ilegible. Mientras que los programas de software libre se publica el código fuente legible para que todo el mundo pueda mejorarlo.

Por ejemplo en la siguiente figura, el programa gráfico **mBlock** que se utiliza en Arduino, mBot, etc... pasa sus instrucciones de lenguaje de programación de bloques parecido a Scratch a lenguaje de **Código de Arduino IDE** y Arduino IDE graba instrucciones binarias de **lenguaje máquina** al Arduino.



## ¡¡A disfrutar!!

Consejo: Te recomendamos visitar el canal de Youtube de Arduinoblocks

<https://www.youtube.com/c/ArduinoBlocks>

Revision #17

Created 1 February 2022 12:51:07 by Equipo CATEDU

Updated 27 December 2022 08:31:17 by Javier Quintana