Software

Las placas electrónicas educativas (Echidna, Micro:bit, Arduino, ESP32 ...) se programan mediante **varias opciones** :

https://docs.google.com/presentation/d/e/2PACX-1vQb1Dv9wN9QK-F6V7yvwDoyzquqwWlGvIyVJr83Yk56kAoYD7bXLnYDm_tCQkeAgg/pubembed?start=falseloop=false&delayms=3000

OPCIÓN LENGUAJE GRÁFICO POR BLOQUES

Recomendado para primaria. Tenemos muchas posibilidades de lenguajes gráficos. Destacamos dos:

- **ECHIDNASCRATCH**
  - **Específico para Echidna e integra la IA CURSO DE ECHIDNA**
- **MBLOCK** Basado en Scratch. Aunque es un programa especializado en el robot comercial mBot, (basado en Arduino), el mismo programa está adaptado para programar Arduino.
  - **CURSO ARDUINO CON MBLOCK** se utiliza Arduino y placa Protoboard
  - **CURSO DE ECHIDNA** se utiliza la Shield Echidnam y EchidnaBlack
  - **CURSO DE MBOT** se utiliza el robot mBot
- **STEAMAKERBLOCKS (antes ARDUINOBLOCKS)** se trabaja online, muy visual y muy amigable. Está adaptado tanto para trabajar tanto Arduino como muchas placas controladoras y en el aula. Podemos verlo en los siguientes cursos:
  - **CURSO ROVER CON ARDUINO** aunque no se utiliza un Arduino, sino un NodeMCU pero la programación es exactamente igual
  - **CURSO DE ARDUINO CON ARDUINOBLOCKS** donde se utiliza el Arduino con una placa protoboard
  - **CURSO ARDUINOBLOCKS EN EL AULA** donde se utiliza la Shield TDR-STEAM

Copyright 2026

  - **CURSO ESP32 EN EL AULA** donde también utiliza la Shiedl TDR Steam pero la placa no es un Arduino, sino ESP32, la programación es exactamente igual.
- **Microbloks** https://microblocks.fun/ placas: Arduino, Microbit, ESP32, RaspberryPico, ...
  ver minitutorial

Otros softwares para programar con bloques

- **Snap4Arduino** https://snap4arduino.rocks/run/ Online, libre... ver compartiva vs mBlock
- **S4A** https://s4a.cat/

EN VIVO ¿Qué es eso?

Existe una posibilidad de utilizar la placa "en vivo" frente a "cargar" el programa en la placa. Es decir, interactuando con el ordenador. El programa está en el PC. En la placa hay un **firmware** que le dice que este a las órdenes del PC. De esta manera podemos por ejemplo:
- Enviar órdenes desde el ordenador a la placa.
Por ejemplo que al pulsar la tecla espacio que se encienda el led D13
- Enviar información desde la placa al ordenador
Por ejemplo que muestre por pantalla nos muestre la cantidad de luz, que registra el sensor LDR, etc...
Que nosotros sepamos, estos programas permiten la programación en vivo :
- **mBlock** placas: Arduino, Microbit, Raspberry Pi, ... robots de Makeblock: mBot, Cyberpi...
-**EchidnaScratch** CURSO DE ECHIDNA
-**Microblocks**
VENTAJAS LA PROGRAMACIÓN EN VIVO PERMITE MUCHO JUEGO Y POSIBILIDADES A LA HORA DE ELABORAR PROYECTOS
INCONVENIENTES: Necesitas el ordenador encendido y conectado al robot.

TAMBIÉN ES EN VIVO PERO ....
Hay otros softwares que técnicamente trabajan en vivo, es decir, que el programa se ejecuta desde el ordenador, no se ejecuta en la placa, son :
- Snap4Arduino para placas Arduino
- **Picobriks** blocks para Picobrick board
Pero no permiten trabajar utilizando los elementos del ordenador (teclado, webcam, pantalla, sprite o objetos,,,)

Es **importante** que entiendas que para trabajar en vivo, la placa tiene que tener cargado un "**firmware**" para que interactúe con el ordenador.

P: ¿Qué es eso de "firmware"?
R: No es más que un software que se graba en los chips de la placa.
P ¿Y por qué se llama así, y no se llama software o programa y en paz?
R: Digamos que como se graba en los chips, es un medio camino entre software y hardware, para diferenciarlo del software habitual.

EN CARGA ¿Qué es eso?

Simplemente el programa que estas haciendo **se carga** en la placa
VENTAJAS: Eres independiente del ordenador, tu robot funciona independiente
DESVENTAJAS Pierdes todas las posibilidades de utilizar los recursos de un ordenador, teclado, pantalla, webcam, altavoces...

Es **importante** que si **cargas** tu programa en la placa, **pierdes** lo que había antes
Es decir, si quieres volver a trabajar EN VIVO tienes que cargar el firmware correspondiente.

OPCIÓN LENGUAJE POR CÓDIGO

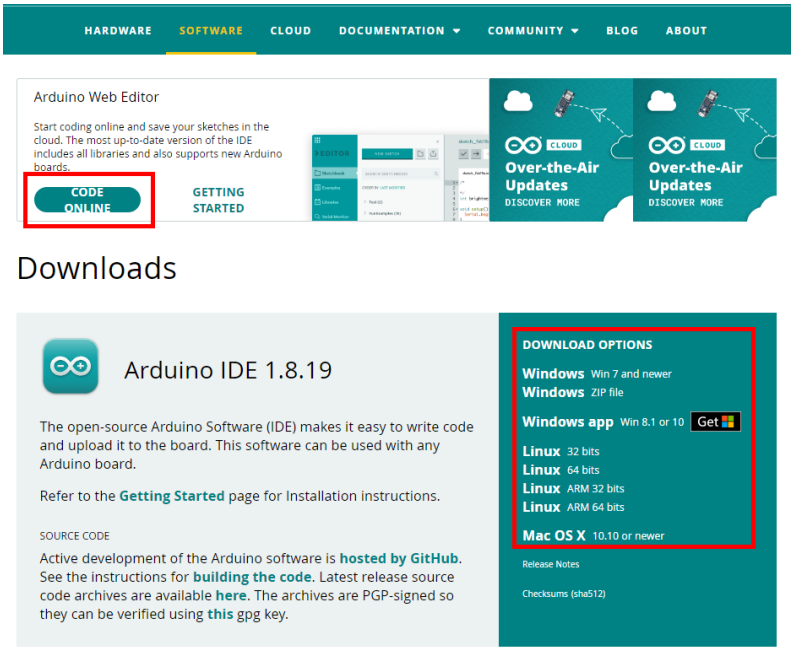
Recomendable a partir de secundaria.

LENGUAJE POR CÓDIGO ARDUINO IDE

Es un lenguaje basado en Wiring y permite la programación del Arduino en un entorno de desarrollo (basado en Processing). El programa se llama **ARDUINO IDE** y se puede descargar desde la página oficial de Arduino: https://www.arduino.cc/en/software.

Hay otra posibilidad que es utilizarlo online, con la ventaja de tener tus proyectos "en la nube" y no depender del equipo. OJO, TIENES QUE TENER INSTALADO EL SOFTWARE **CREATE AGENT**

https://create.arduino.cc/getting-started/plugin/welcome



En los cursos de CATEDU se ha utilizado el lenguaje por código empezando desde cero en:

- **CURSO ARDUINO CON CÓDIGO** donde se trabaja con el Arduino y con diferentes sensores y actuadores, con o sin placa Shield Edubásica.
- **CURSO DE DOMOTICA CON ARDUINO** donde se realiza una maqueta de una casa controlada con domótica. También el curso ofrece la versión de hacer la misma maqueta utilizando lenguaje gráfico por bloques.

Recomendamos estas hojas resumen si vas a trabajar con código:

- En Español: enlaceDrive, enlaceGithub
- En Inglés: enlaceDrive, enlaceGithub, enlaceSpakrfun

LENGUAJE POR CÓDIGO PYTHON

Es un lenguaje más amigable que ArduinoIDE y tiene muchos campos de aplicación aparte de la robótica

- Arduino tiene su propia versión para trabajar con sus placas compatibles: **Arduino Lab for Micropython**
  - Ver curso Arduino ALVIK
- Un compilador universal **Thonny**
  - Ver curso Pico-briks

Copyright 2026

VENTAJAS E INCONVENIENTES
LENGUAJE GRÁFICO POR BLOQUES vs CÓDIGO

El lenguaje gráfico por bloques es un lenguaje **sencillo de utilizar**, nos evita tener en cuenta muchas **librerías** y **cálculos.**

> Otra ventaja, es que el lenguaje por bloques es el único que permite programación "en vivo"

Por ejemplo, la instrucción leer valor distancia el sensor ultrasonidos, mediante programación por bloques es

mientras que en código es

```
double distancia;

double fnc_ultrasonic_distance(int _t, int _e){
  unsigned long dur=0;
  digitalWrite(_t, LOW);
  delayMicroseconds(5);
  digitalWrite(_t, HIGH);
  delayMicroseconds(10);
  digitalWrite(_t, LOW);
  dur = pulseIn(_e, HIGH, 18000);
      // devuelve cuanto tarda el pulso alto en microseg; 18000 es el tiempo a esperar limite
  if(dur==0)return 999.0;
  return (dur/57);
      // la velocidad del sonido es 344m/s = 34400 cm/seg = 0,0344 cm/microseg
     // como v=e/t luego e = v*t y como cuenta la ida y la vuelta distancia = v*t/2
     // luego distancia = 0,0344/2 * dur = dur/57
}
```

Copyright 2026

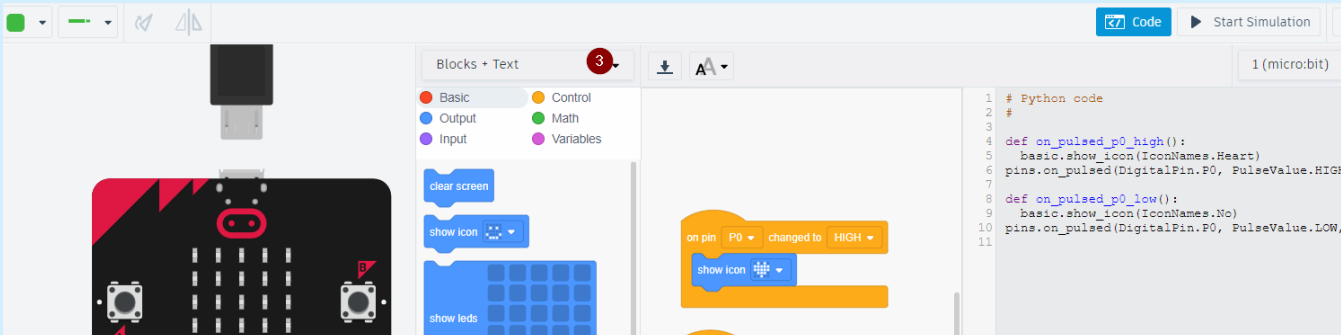
```
void setup()
{
  pinMode(6, OUTPUT);
pinMode(5, INPUT);

}


void loop()
{

  distancia = fnc_ultrasonic_distance(6,5);

}
```

Como se puede ver en **código**, tiene que calcular la distancia haciendo cálculos del tiempo de rebote del eco, mientras que el **gráfico** es sumamente sencillo y se centra en el objetivo del algoritmo a crear, no en lo accesorio. Esto hace que un lenguaje gráfico por bloques se puede aplicar **desde los 8 años**.

Por otra parte, el lenguaje **código tiene todo el potencial,** es decir, no todo está en los lenguajes gráficos. Si se quiere cosas más avanzadas, hay que recurrir al código.

> Un lenguaje **gráfico** se convierte en lenguaje **código**, pero **al revés no se puede,** debido a que el código es más depurado y no tiene la información necesaria para volver a su origen en bloques, ya lo has visto con el anterior ejemplo, el código tiene más información.

> **¿No te lo crees?** Haz la prueba, métete en https://www.tinkercad.com/ crea un programa con bloques, dale a la pestaña de código y te aparecerá una advertencia que perderás el programa con bloques ! **no puedes volver atrás**!
>
> Curiosamente, tiene una opción *bloques+código* que traduce cada bloque con un código, es decir, traduce cada bloque sin perder información, sólo de esa manera se puede pasar de bloques a código y viceversa.



> Hay herramientas para pasar de bloques a código pero no al revés

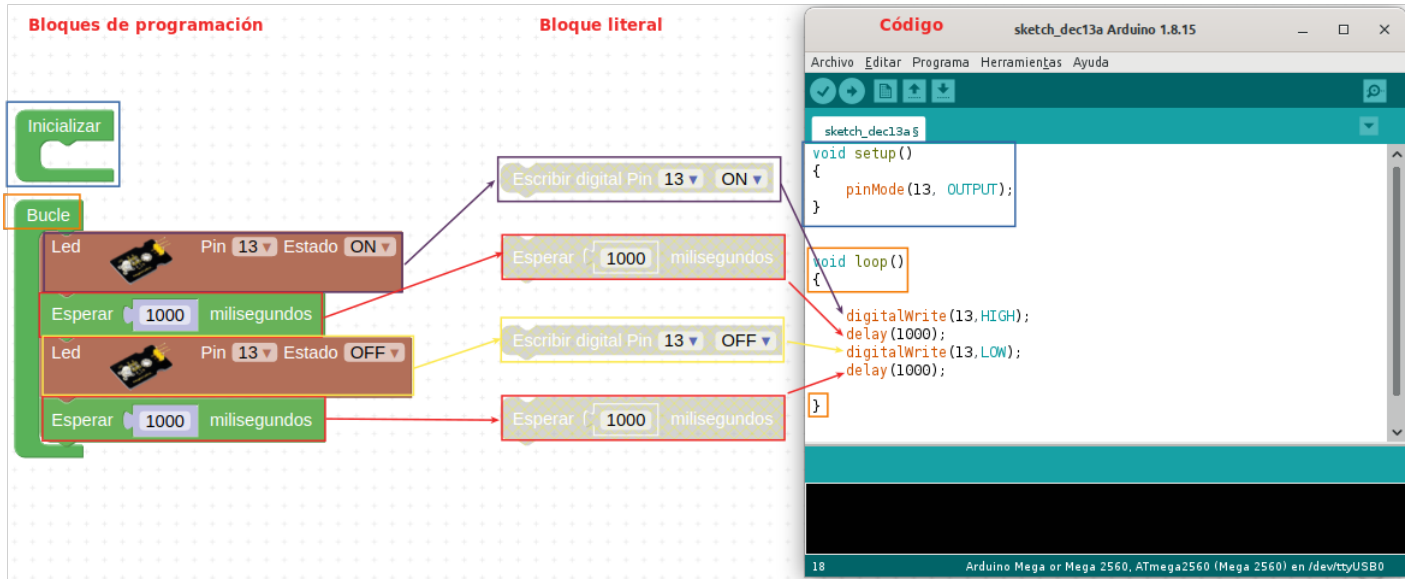


Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

> El lenguaje **código** se traduce en lenguaje **máquina** (ceros y unos) entendible para el microprocesador, **pero al revés no se puede**.

En este vídeo, en mi opinión se olvida de mBlock, Snap4Arduino, S4A pero puedes ver un vistazo de los diferentes editores

OPCIÓN SIMULACIÓN

Incluimos dentro del apartado de Software los diferentes programas que hay para simular placas electrónicas como Arduino, ESP32, Micro:bit etc...

Tinkercad

Esta herramienta https://www.tinkercad.com aparece en el **Curso Arduino con código** en la práctica Comunicación entre dos Arduinos, pero también es una plataforma que sirve para hacer los diseños de elementos 3D, ver curso **Impresión 3D con Tinkercad**

Tiene la ventaja que es aplicación **online**, muy **visual** y buscan un reflejo de la práctica **real,** además de estar la herramienta **adaptada al aula** (gestión de alumnos y proyectos). Como desventajas podemos decir que no tiene mucha variedad de componentes electrónicos y la simulación es algo lenta.

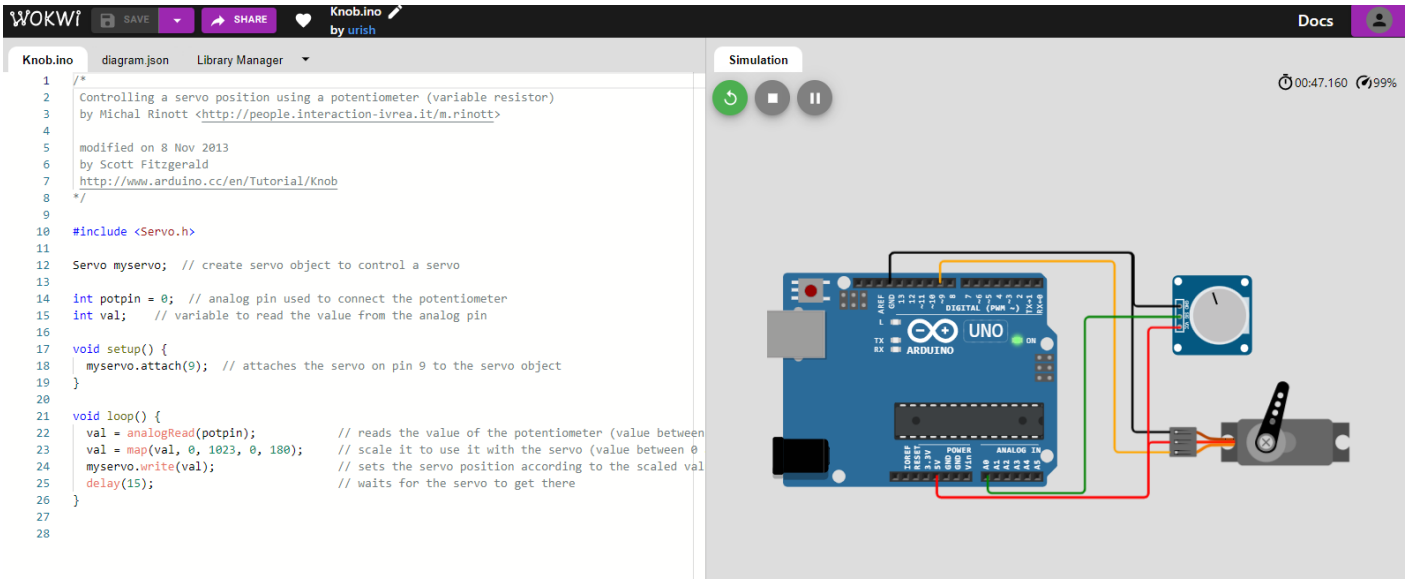
https://www.youtube.com/embed/pXEv0wxW9Jo?rel=0

Además permite embeber, **pruébalo !!!** □□□□□ Pulsa **Iniciar simulación** y luego pulsa el **botón A** de cualquiera de los dos micro:bits

https://www.tinkercad.com/embed/8vBL2E8wWDx?editbtn=1

Wokwi

Si Tinkercad se queda corto, puedes probar esta plataforma https://wokwi.com/ con muchas posibilidades. Es **online** y puede trabajar con **multitud de placas**: ArduinoUno, ESP32, Raspberry,,,,
Como única desventaja que encontramos, es que echamos de menos la realidad de Tinkercad, por ejemplo no puedes poner una placa protoboard para realizar las conexiones, pero a cambio se gana simplicidad de cableado.



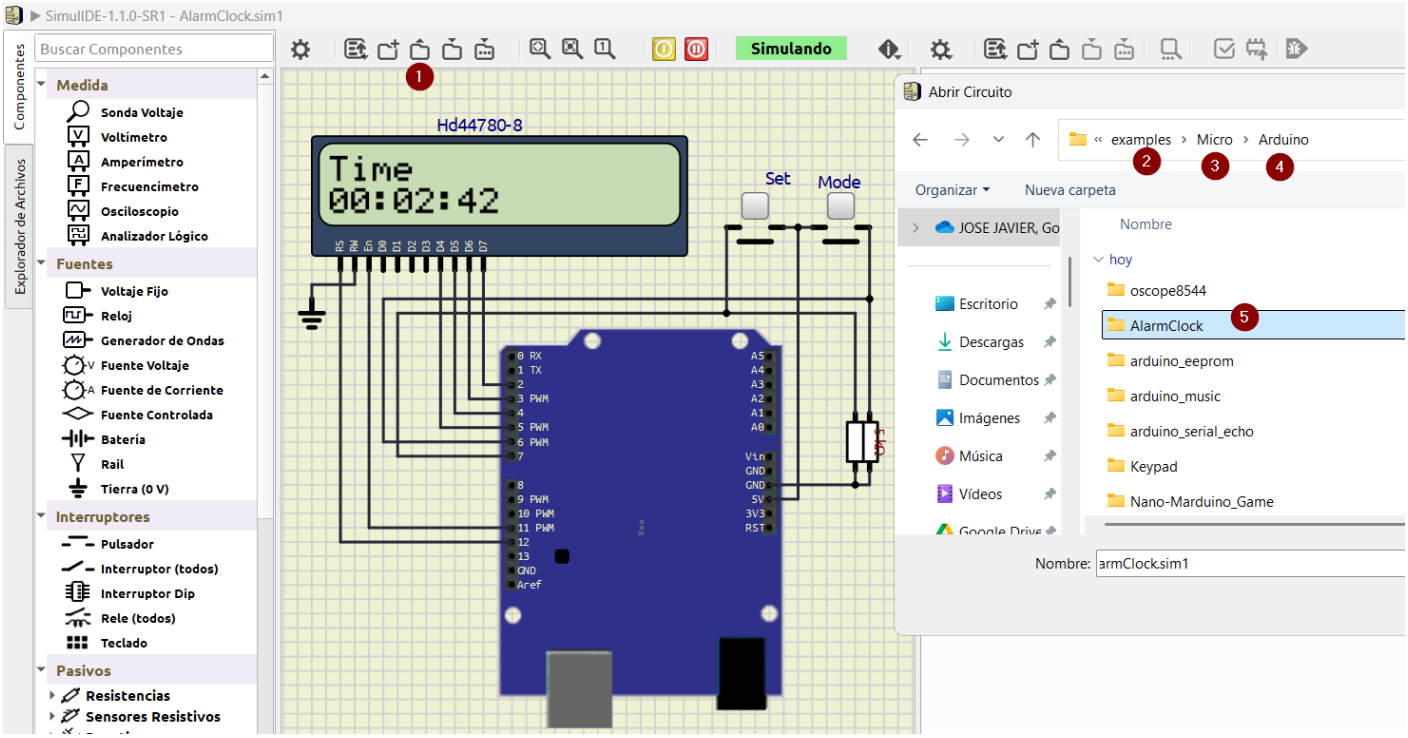
UnoArduSim

Es una **aplicación local.** **UnoArduSim** además es una aplicación portable fácil de instalar y con los elementos de leds, motores servos ya preparados, ideal para ejemplos sencillos y para examinar señales y no depender de Internet, pero no es tan versátil.

https://www.youtube.com/embed/WLJ_l4uGjXg?rel=0

SIMULIDE

En https://simulide.com/ podemos encontrar un programa local de software libre genérico para electrónica, incluido Arduino. En esta captura se puede ver uno de los ejemplos que incorpora el programa:



OPCIÓN SÓLO DIBUJAR

- **TinkerCad** es un buen programa para dibujar los planos
  - ✓ permite también la simulación
  - ✓ permite embeber y compartir
  - ✗ no tiene muchos componentes
- **SimulIDE** es software libre. Es un programa portable.
  - ✓ Tiene muchos componentes
  - ✓ permite también la simulación
  - ✗ le faltan algunos sensores, pero van incorporando
- **Fritzing** es un clásico. Es un programa portable.
  - ✓ Tiene muchos componentes
  - ✗ no es gratis, hay que pagar 8€
- **Circuit canvas**

- puede compartir por ejemplo
- tiene buenos tutoriales sobre electrónica
- todo en inglés

Revision #73
Created 2022-10-28 13:23:13 CEST by Javier Quintana
Updated 2026-01-16 08:18:47 CET by Javier Quintana

Copyright 2026