Qué es la progrmación por bloques

- ¿Qué es la programación por bloques?
- Code y Scratch

Copyright 2026

¿Qué es la programación por bloques?

La programación por bloques nos permite **programar de forma gráfica**, sin necesidad de conocer la sintaxis de un lenguaje de programación. Es ideal para comenzar a programar para el alumnado, puesto que permite **descomponer el problema de la programación** en dos subproblemas, y simplificar mucho uno de ellos:

- Saber la **lógica** de la programación: esto supone entender el uso de entradas, salidas, estructuras de almacenamiento de datos y operadores.
- Saber el **lenguaje** de programación: traducir el programa a un lenguaje que el ordenador pueda entender. Esto es lo que nos facilita la programación gráfica por bloques.

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

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);
```

Copyright 2026

```
delayMicroseconds(10);
digitalWrite(_t, LOW);
dur = pulseIn(_e, HIGH, 18000);
if(dur==0)return 999.0;
return (dur/57);
}

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.

Copyright 2026

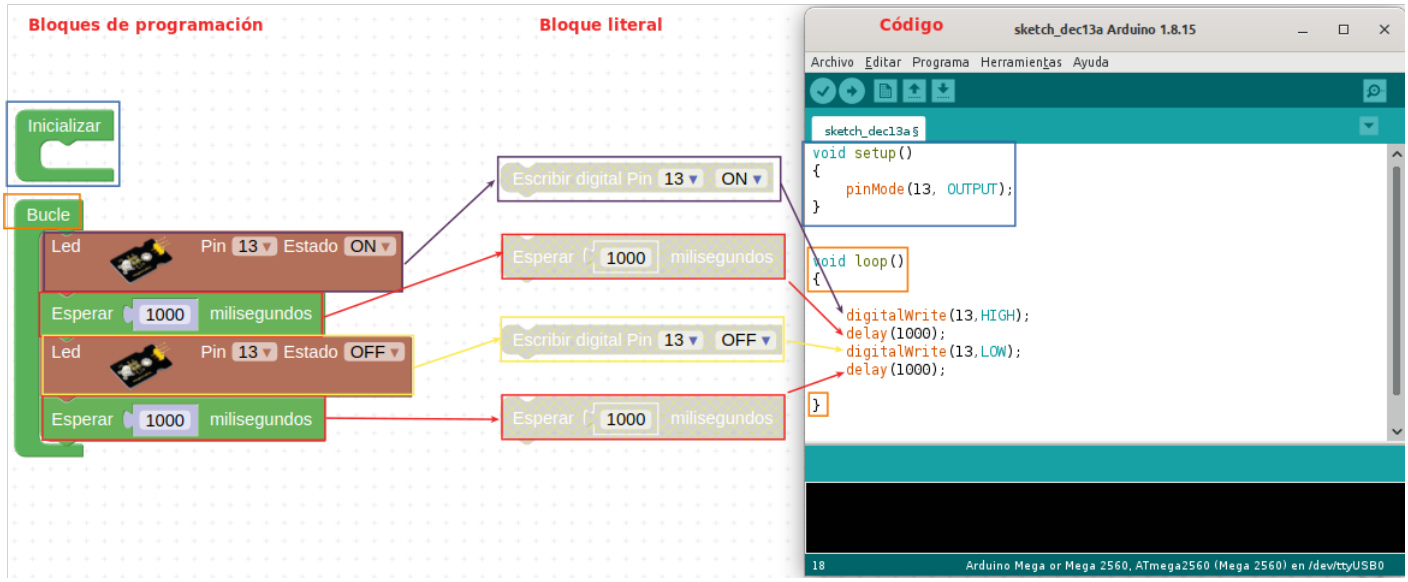


Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Una gran ventaja de la programación por bloques es que muchos de los programas que la utilizan son software libre o **han basado su código fuente en el mismo código, que es software libre** y se encuentra liberado por sus autores. Esto se traduce en que nuestro alumnado al aprender una herramienta concreta se **alfabetiza y se capacita para muchas otras** sin saberlo.

Programación gráfica o por bloques



> Más información sobre la programación por bloques aquí

Code y Scratch

Dos de los entornos de programación por bloques más extendidos y pioneros a nivel mundial son **Scratch y Code.**

En capítulos posteriores de este curso conocerás más en detalle cada uno de ellos. No obstante queríamos comenzar con un **análisis de fortalezas y debilidades de cada uno** de ellos que quizás pueda ayudarte a la hora de decidir a posteriori cual utilizar con tu alumnado según tu contexto.



1. **FACILIDAD DE USO:** Si bien ambas plataformas son fáciles de usar, creemos que code.org está más orientado a usuarios principiantes y jóvenes. Los cursos están bien estructurados en función de los grupos de edad. Ayuda a los niños a construir una base sólida para ayudarlos en su viaje de codificación. Entonces, para los usuarios principiantes, es una victoria para **code.org.**
2. **ESTRUCTURACIÓN DEL APRENDIZAJE:** Los programadores nuevos pueden perderse en el mundo abierto de scratch y perder el interés, mientras que code.org tiene contenido estructurado y progresivo disponible según la edad y los niveles de conocimiento de los niños. Para esta ronda volvemos a optar por **code.org** como ganador debido a sus cursos personalizados creados de acuerdo con la edad y los niveles de conocimiento de los niños.
3. **APRENDIZAJE AUTÓNOMO:** Code.org es excelente para usuarios principiantes, pero una vez que comprenden los conceptos básicos, el contenido del curso en code.org se agota con bastante rapidez y la herramienta en sí tiene muchas limitaciones. Por otro lado, debido a la forma en que está diseñado Scratch, uno puede pasar horas y horas aprendiendo y dominando los conceptos sin agotar el contenido o aburrirse. Por lo tanto en este aspecto nuestro ganador es **Scratch**.

Copyright 2026

4. **FORMATO MÁS FLEXIBLE Y MAYOR PROMOCIÓN DE LA CREATIVIDAD:** Las personas codificadoras en el mundo real deben lidiar con mucha ambigüedad y, una vez aprendidos, los conceptos se aplican de forma aleatoria. Si bien existe una estructura básica de sintaxis y el algoritmo debe seguir un patrón, existen múltiples formas de resolver el mismo problema. Por lo tanto, es necesario aprender a lidiar con la ambigüedad. En este sentido, **Scratch supera claramente a code.org.** A diferencia de code.org, donde cada lección se presenta en detalle, en Scratch el alumnado aprende a trabajar los conceptos en un formato más abierto. Creemos que esto ayuda a lidiar con la ambigüedad y las múltiples formas de abordar un problema de codificación.

CONCLUSIÓN: Dados los puntos anteriores, nuestra recomendación es que, para programadores principiantes, **comiencen aprendiendo los conceptos en code.org** y luego continúen con **Scratch para obtener un aprendizaje avanzado**. Una vez que haya practicado en Scratch y tenga un buen manejo de los conceptos básicos, el alumnado estará preparado para pasar a otros lenguajes de programación basados en texto como JAVA o Python.

> La información de esta página ha sido extraída y traducida de https://codechamp.ca/scratch-vs-code-org-which-is-best-for-your-kids-to-learn-to-code/

Copyright 2026