suma+ 90
Artículos
Marzo 2019
pp. 17-30

Introducción a la programación con Scratch*

Virgilio Gómez Rubio
María José Haro Delicado
Francisco Palmí Perales

En este artículo describimos el entorno de programación Scratch y cómo puede ser utilizado en el aula. Entre las muchas características de este software libre, destacamos su entorno interactivo y visual, que facilita la creación de programas a los alumnos. Los programas operan sobre una serie de imágenes ubicadas en un entorno. La creación de programas se realiza mediante el ensamblaje de bloques que representan operaciones sencillas, como mover la imagen, emitir un sonido, etc., de manera que los alumnos pueden crear sus propias historias.

Palabras clave: Scratch, Programación, Algoritmo, Entorno visual, Lenguaje de programación.

Introduction to programming with Scratch

In this paper we describe the programming language Scratch and how it can be used in the classroom. Among its many characteristics, we highlight its interactive and visual environment, which makes it easy for the student to write programmes. Programmes operate on a number of images within a stage. These programmes are created by assembling different blocks which represent simple operations, such as moving the image, making a noise, etc., in a way that allows students to develop their own stories.

Keywords: Scratch, Programming, Algorithm, Visual environment, Programming language.

Scratch es un lenguaje de programación visual que permite a niños y jóvenes adentrarse en el terreno de la programación y familiarizarse con conceptos clave de las ciencias de la computación como si de un juego se tratase. Funciona a base de piezas o bloques que representan distintas operaciones y que se encajan como en un puzle. Mediante el desarrollo de pequeños *programas* que aumentan su complejidad de manera natural según se avanza, los estudiantes son capaces, no solo de aprender conceptos y procedimientos propios del terreno de la computación, sino que, además, adquirirán habilidades y serán capaces de desarrollar estrategias que les serán útiles en cantidad de situaciones cotidianas.

Su interfaz gráfica y atractiva permite interactuar desde el principio y obtener resultados llamativos con muy pocos conocimientos de programación. Siguiendo la lógica y la intuición, los estudiantes podrán desarrollar pequeños programas y aplicaciones, lo que constituirá una fuente de motivación importante y favorecerá la creatividad y el autoaprendizaje. La práctica permitirá progresar desarrollando aplicaciones más depuradas desde el punto de vista sintáctico y con estructuras más complejas, en las que aparezcan conceptos avanzados, como el de recursividad. Por otro lado, se promueve el aprendizaje colabo-

Artículo recibido en *Suma* en enero de 2018 y aceptado en julio de 2018

rativo, ya que los chicos pueden compartir sus trabajos y programas con sus compañeros de clase. A la vez, se les prepara para el futuro en un mundo que demanda, cada vez más, gente preparada en el terreno de la programación y la computación.

¿Por qué es importante que los niños tomen contacto con la programación desde muy jóvenes?

Los niños, lo queramos o no, son grandes consumidores de tecnología. Desde muy pequeños manejan móviles, tablets e incluso ordenadores con total naturalidad y vale más que sean productores activos que consumidores pasivos de tecnología. La tecnología está dando forma al mundo actual y lo hará todavía más en el futuro resolviendo gran variedad de problemas en todo tipo de situaciones y áreas. Programando, los niños desarrollarán habilidades propias del pensamiento computacional como lo son el pensamiento lógico y la capacidad de abstracción; aprenderán a llevar a cabo tareas de diversa índole de una manera clara y precisa, dividiendo los problemas en partes que se resuelven de una en una y después se ensamblan para alcanzar un objetivo; aprenderán a detectar y corregir errores y a descubrir patrones. Estas son habilidades transversales necesarias en cualquier terreno cotidiano en que nos movamos. El objetivo no es que los niños aprendan a programar, sino que aprendan programando.

Antecedentes

Tomar contacto con la programación desde edades muy tempranas no es algo nuevo. Encontramos antecedentes en el uso de lenguajes de programación como Logo, que durante los años 70 y 80 enseñó a niños, y también a mayores, a dar sus primeros pasos en el mundo de la informática y de la programación. Las características de este lenguaje lo hacían particularmente útil en el aprendizaje de las matemáticas y, más concretamente, de la geometría. Se planteaban retos que tenían que resolverse mediante el desarrollo de pequeños programas. La tortuga se convertía en una poderosa aliada que animaba a pensar y a detectar y depurar errores. La figura 1 muestra un ejemplo del entorno de programación proporcionado por Logo, en el que se puede apreciar la tortuga que realizará las acciones programadas por el usuario.

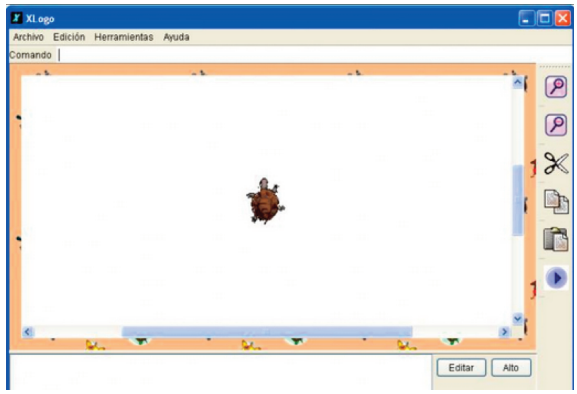

Figura 1. Ejemplo de sesión
del entorno de programación Logo

¿Por qué Scratch?

Scratch ha sido desarrollado por el grupo Lifelong Kindergarten del MIT (Instituto Tecnológico de Massachusetts, EE.UU.) con el único propósito de enseñar a los niños a programar. Se puede trabajar con Scratch online, o bien descargándolo de la dirección <https://scratch.mit.edu>.

Entre las ventajas que ofrece Scratch para trabajar con niños y adolescentes se encuentran:

- Se descarga con facilidad y se puede trabajar con él en cualquier sistema operativo.
- Es un lenguaje de programación libre con el que se pueden crear historias interactivas, juegos y animaciones.
- Los programas operan sobre objetos llamados *personajes* (representados por imágenes) que interactúan entre ellos en un entorno llamado *escenario*. Poder ver actuar a los personajes en el escenario ayuda a comprobar en el acto el funcionamiento

Figura 2. Ejemplo de *personaje* disponible en Scratch

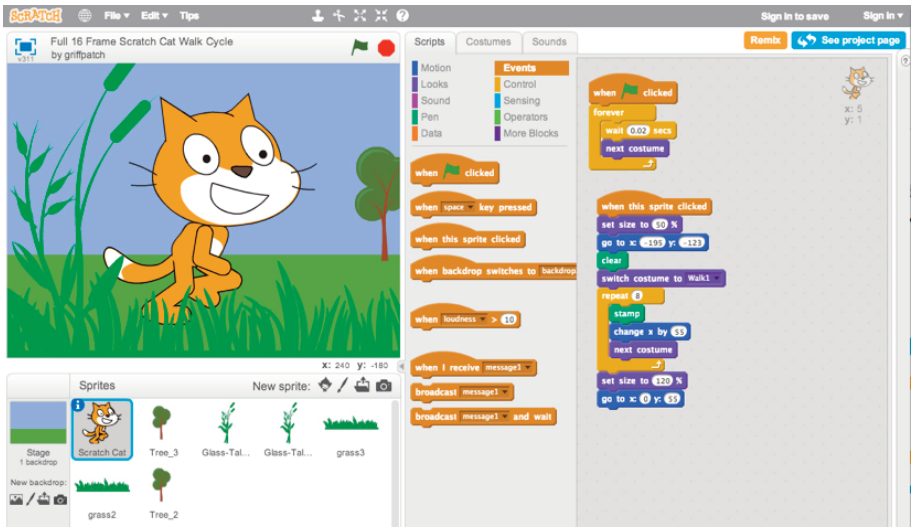

Figura 3. Ejemplo del entorno de programación de Scratch con varios *personajes* en el *escenario* (izquierda) y los bloques del programa desarrollado (derecha)

de las instrucciones y a depurar errores cuando los resultados no son los esperados. La figura 2 muestra la imagen del *personaje* que aparece por defecto en Scratch, aunque es posible elegir muchos otros.

— Es un lenguaje visual. Los programas se crean arrastrando y soltando bloques, como si de un puzle se tratara, en vez de usar comandos. Algunos de los errores se detectan inmediatamente al intentar encajar bloques que no deberían unirse. La imposibilidad de cometer cierto tipo de errores ayuda a centrar la atención en la resolución del problema. La figura 3 muestra un ejemplo del entorno de Scratch en el que se pueden ver varios *personajes* en el *escenario* y un ejemplo de programa construido a base de bloques con distintas operaciones.

— El entorno es multimedia, lo que quiere decir que hay bloques que sirven para producir sonidos, usar el teclado o el ratón o incluso usar la *webcam*. De esta manera se pueden crear historias animadas de forma fácil y rápida.

— Los usuarios pueden compartir sus creaciones con otros miembros de la comunidad Scratch, comentar el trabajo de los demás y recibir sus comentarios y usar las ideas y código de otras personas, como fuente de inspiración en el desarrollo de sus propias creaciones.

Dos principios básicos

A la hora de presentar el uso de Scratch a los alumnos trabajaremos con dos conceptos fundamentales:

— Algoritmo: Es la secuencia ordenada de pasos que necesitamos seguir para resolver un problema.

— Codificar: Después de desarrollar el algoritmo hay que escribirlo en un lenguaje que el ordenador entienda. A ello se le llama codificar.

Además, al proceso de crear algoritmos y codificarlos se le llama programar.

Describiendo el entorno

Para trabajar con Scratch tenemos dos opciones:

— Descargar el programa y trabajar sin conexión.

— Ir directamente a la página web de Scratch <https://scratch.mit.edu> y trabajar usando el editor on-line.

Si optamos por esta segunda modalidad solo necesitamos hacer clic en Crear para comenzar una nueva sesión y desarrollar un programa nuevo, tal y como puede apreciarse en la figura 4, que muestra la web de inicio de Scratch.

Figura 4. Editor on-line de Scratch disponible en <https://scratch.mit.edu>

La ventaja de trabajar de esta segunda forma es que tendremos acceso a nuestros proyectos en cualquier lugar en el que dispongamos de conexión a Internet. De todas formas, si preferimos descargarlo podemos ir a la parte final de la página y pinchar en Editor sin conexión, tal y como se puede ver en la figura 5.

Figura 5. Pie de la web de Scratch desde la que se puede descargar el editor de Scratch pinchando en Editor sin Conexión

Después de pinchar en Crear, nos encontraremos un entorno como el que aparece en la figura 6. El área de la izquierda es el escenario y ahí es donde podremos observar los efectos del programa que realicemos. Además, aparece un menú en el que seleccionar el fondo del escenario y el *personaje* a utilizar. Durante el desarrollo de programas es posible utilizar varios *personajes* e incluso cambiar su apariencia.

En la parte de la derecha es donde podemos encontrar los distintos bloques con los comandos de programación, los diferentes *personajes* y sonidos disponibles, así como un área en la que poder combinar los distintos bloques para elaborar el programa. Estos bloques están divididos en grupos

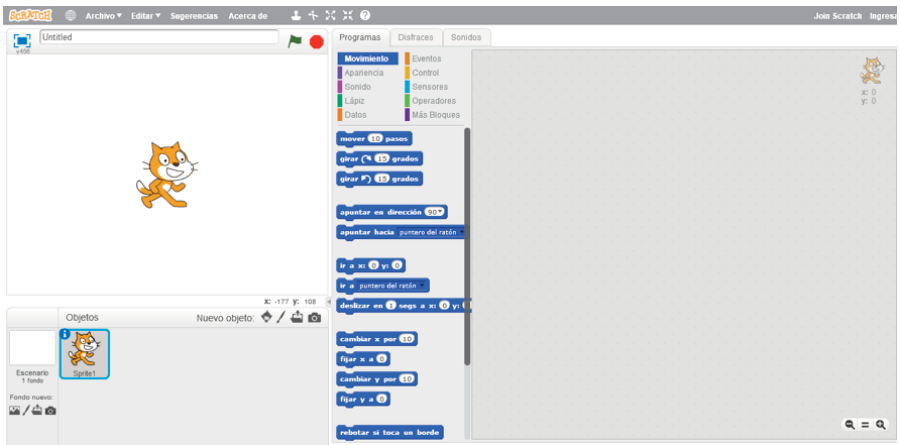

Figura 6. Inicio del editor de Scratch

en un menú, dependiendo de su función, y cada grupo aparece representado por un color distinto dependiendo de su funcionalidad. Por ejemplo, los bloques que se refieren a los comandos de movimiento del *personaje* aparecen representados en azul, los que tienen que ver con la apariencia del *personaje* en violeta, etc.

Podemos cambiar el escenario blanco por diferentes fondos que podremos obtener de la biblioteca de Scratch o que crearemos nosotros. Vamos a pinchar en la Librería y a seleccionar uno de ellos, tal y como aparece indicado en la figura 7.

Figura 7. Pinchando sobre el icono de escenario se puede acceder a la biblioteca de fondos para el *escenario* de Scratch y escoger otro más apropiado a nuestras necesidades

De los muchos que hay, vamos a escoger, por ejemplo, «school 1» que se refiere a un fondo que representa la parte delantera de una escuela. Podremos observar que la situación cambia en la imagen de la figura 8.

El escenario mide 480 unidades de ancho y 360 de alto. Está dividido en una cuadrícula representada por puntos de coordenadas (x, y). Las coordenadas del punto central son (0, 0). Para averiguar la posición de un punto dentro del escenario basta con pasar el ratón por encima y

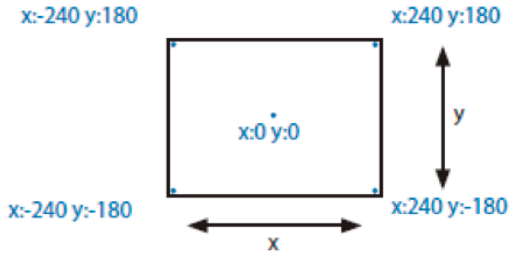

Figura 9. Representación de las dimensiones del escenario de Scratch, así como de las coordenadas de sus límites

ver las coordenadas que se muestran en la parte inferior derecha del escenario. La figura 9 muestra las dimensiones del rectángulo que define el escenario y de las coordenadas de los cuatro puntos que marcan sus límites. El uso de estas coordenadas puede ser útil, por ejemplo, a la hora de ubicar o desplazar el *personaje* durante el desarrollo de un programa.

Como ya se ha mencionado con anterioridad, a los objetos que creemos con Scratch se les llama *personajes*. Al empezar, nos encontraremos con la mascota de Scratch (un gato, que aparece en la fgura 2), pero también podemos crear nuestros propios objetos o seleccionarlos de nuevo de la biblioteca de Scratch pinchando en el símbolo que aparece indicado en la figura 10.

Figura 10. Pinchando en el símbolo señalado podemos seleccionar el *personaje* que usaremos en el programa

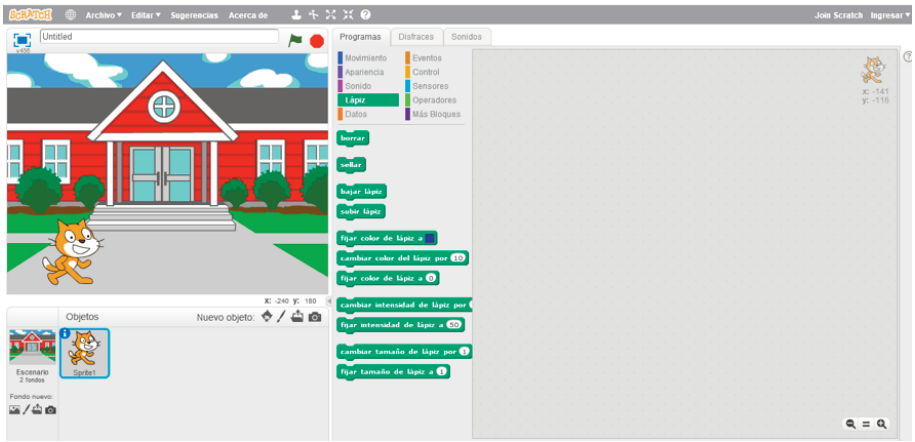

Figura 8. Entorno de Scratch al cambiar el fondo del *escenario* usando el llamado «school 1»

Vamos a controlar los movimientos, y el comportamiento en general, de nuestro objeto arrastrando y soltando bloques que están organizados por categorías: **Movimiento, Apariencia, Sonido, Lápiz, Datos, Eventos** (para indicar cuando se deben ejecutar determinados bloques de código), **Control, Sensores, Operadores, Más bloques** (para crear nuestros propios bloques).

Figura 11. Listado de los grupos de bloques de comandos disponibles en Scratch

Para actuar sobre un objeto, hacemos clic sobre él y a continuación arrastramos bloques al área de código (parte derecha de la figura 6), conectándolos entre sí. Generalmente, empezaremos con el bloque de inicio de programa, que aparece en la figura 12. Este bloque hace que se inicie la ejecución de un programa al pinchar la bandera verde que aparece en la parte derecha del entorno de Scratch.

Figura 12. Bloque que marca el inicio de un programa

Si se comete un error en la creación del programa, es posible separar los bloques en el área de código y arrastrarlos al área de bloques (parte central en la figura 6) o se hace clic con el botón derecho sobre el bloque que se quiere borrar, marcando borrar en el cuadro de diálogo que se despliega.

La figura 13 muestra un ejemplo de programa en el que se han seleccionado dos bloques de movimiento. El primero hace girar el *personaje* 90 grados, mientras que el segundo le hace avanzar 40 pasos. Estos valores se pueden modificar pinchando sobre ellos en los bloques que se han colocado en el programa.

Hay *disfraces* para los objetos, lo que quiere decir que podemos cambiar su apariencia cuando actúan sobre el escenario. La figura 14 muestra un programa más complejo en el que el *personaje* cambia de aspecto y realiza una serie de acciones más, como moverse, decir «Buenos días», cambiar de posición, volver a cambiar de aspecto y emitir un maullido. Para hacer que el programa se ejecute, presionaremos sobre la bandera verde.

Figura 14. Pequeño programa para cambiar la apariencia de un *personaje* y que realice varias acciones más

Si queremos guardar nuestro proyecto, le asignaremos un nombre y haremos clic sobre **descargar a tu computadora** (ver figura 15).

Si estás registrado, en el menú archivo aparecerán más opciones. En este caso, haz clic en **Guardar ahora**, tal y como aparece en la figura 16.

Figura 13. Ejemplo de programa para girar el *personaje* 90 grados y hacerlo avanzar 40 pasos

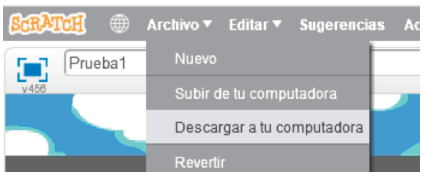

Figura 15. Menú para la descarga del programa realizado cuando se usa el editor on-line

Figura 16. Opción de guardar un trabajo on-line para usuarios registrados

Al guardar de esta última forma, el proyecto se guarda en Mis cosas. Podemos acceder a él desde Archivo > Ir a mis cosas (ver figura 17).

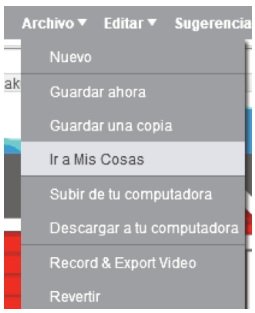

Figura 17. Acceso a trabajos guardados para usuarios registrados

Si se hace de esta forma, podremos compartir nuestro proyecto y cualquier miembro de la comunidad Scratch podrá verlo. También podremos nosotros acceder a otros proyectos e incluso ver su código para ayudarnos en nuestro trabajo.

Más sobre bloques

Como ya se ha comentado, los bloques están agrupados en 10 grupos distintos que podemos identificar por el color (ver figura 11). Hablaremos muy brevemente de cada uno de dichos grupos:

- Grupo de movimiento: Son instrucciones que nos permitirán mover al objeto, girarlo en el sentido de las agujas del reloj y en el contrario, colocarlo en posiciones concretas, determinadas por coordenadas, rebotar al tocar un borde, etc.
- Grupo de apariencia: Estas instrucciones permiten cambiar la apariencia escogiendo nuevos disfraces, aumentando o disminuyendo, de una o gradualmente, el tamaño inicial del objeto, poniendo el objeto en la capa especificada, estableciendo efectos de color, escondiendo, enviando al frente, etc.
- Grupo de sonido: Permiten tocar sonidos, establecer el instrumento, cambiar el volumen, cambiar el tempo, etc.
- Grupo de lápiz: Permite bajar y subir lápiz, borrar, fijar el color del lápiz, cambiar la intensidad, el grosor del trazo, etc.
- Grupo de datos: Nos permite crear y trabajar con variables.
- Grupo de eventos: Contiene el bloque de inicio (figura 13) y, en general, bloques que indican acciones que se desencadenarán cuando se den determinadas situaciones como el aumento de un sonido, presionar el espaciador, recibir o enviar un mensaje, al hacer clic en un objeto, cuando el fondo cambie, etc.
- Grupo de control: Con estas instrucciones se detectan acciones realizadas por otros objetos y se actúa ante ellos. También se ejecutan determinadas acciones según se presione o no una determinada tecla. Procesos iterativos y recursivos se pueden llevar a cabo usando las instrucciones de este bloque, ya que es posible repetir acciones o actuar de una manera u otra por medio de los condicionales.
- Grupo de sensores: Estas instrucciones permiten detectar si hay teclas presionadas, si los objetos están tocando bordes o colores determinados, se puede leer la posición del ratón, medir la distancia al puntero del ratón, preguntar algo y esperar la respuesta, que se almacenará en una variable, etc.
- Grupo de operadores: Las instrucciones de este bloque permiten realizar operaciones

lógicas, operaciones matemáticas básicas como sumar, restar, multiplicar o dividir, calcular raíces cuadradas, y generar números al azar entre otras cosas.

— Grupo Más bloques: Permite crear nuevos bloques y extensiones que representen comandos de programación nuevos construidos a partir de otros más sencillos.

Algunos ejemplos

Presentándonos

Imaginemos que dos personas se encuentran sobre el escenario y se saludan preguntándose cómo se llaman.

Seleccionemos primero un escenario, haciendo doble clic sobre Escenario nuevo. Se abrirá una página como la de la figura 18, y en ella seleccionaremos el escenario que más se adapte a lo que queremos.

A continuación, seleccionamos y añadimos al escenario los dos objetos que van a representar el papel de personas que se encuentran y saludan. Esto lo haremos haciendo doble clic en Nuevo objeto (ver figura 10) y seleccionando los que más nos gusten. Nosotros hemos seleccionado los de un chico y una chica. Dentro de la pantalla que se abre, utilizaremos el código de la figura 19 para cada uno de los objetos.

Podemos ver el resultado de la ejecución de este programa en la figura 20. Este programa está disponible en <https://scratch.mit.edu/projects/173539313>.

Figura 19. Programas para que se saluden los dos *sprites*. Un *personaje* ha de usar el primer programa, mientras que el otro usará el segundo código

Figura 20. Resultado del programa realizado en el que dos *personajes* se saludan mutuamente

Figura 18. Ejemplos de fondos para el *escenario* de Scratch

Jugando con el lápiz

Vamos a ver ahora otro ejemplo con el que podemos practicar con el lápiz para realizar dibujos sobre el escenario. Primero seleccionaremos un *personaje* con forma de lápiz (o como mejor nos parezca) y después dibujaremos con él un cuadrado. Vamos a hacerlo de manera que antes de empezar a dibujar, el lápiz se sitúe en el centro del escenario y que cuando termine de hacer el cuadrado vuelva a la posición original y con la misma orientación, dispuesto para volver a dibujar otro cuadrado. Empezará a hacer el dibujo cuando hagamos clic sobre el objeto. Al terminar su tarea, y después de esperar breves segundos, el dibujo desaparecerá. Repetiremos el dibujo indefinidamente, cambiando cada vez el color del lápiz.

El programa quedaría tal y como aparece en la figura 21.

Se puede observar lo que ocurre en el enlace <https://scratch.mit.edu/projects/173595491> (para que el lápiz inicie su movimiento hay que pinchar sobre el objeto).

Figura 21. Programa para dibujar un cuadrado

Si nos fijamos un poco en el programa de la figura 21, hemos repetido lo mismo cuatro veces. Podemos usar las opciones de repetición del programa para evitar usar tantas instrucciones. Podemos ver el programa modificado en la figura 22.

Figura 22. Programa para dibujar un cuadrado con la opción de repetir una parte del código cuatro veces

Se puede ver cómo funciona el código en <https://scratch.mit.edu/projects/173604146/>.

Por último, el resultado de este programa se puede ver en la figura 23.

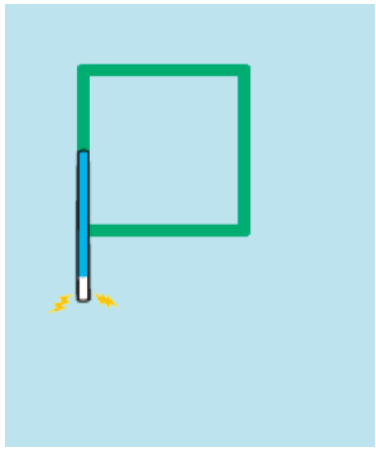

Figura 23. Resultado del programa que dibuja un cuadrado

¿Cómo se podría hacer un octógono? Un programa para hacerlo con Scratch está disponible en <https://scratch.mit.edu/projects/173624461/>.

Preguntas y respuestas

Vamos a ver a continuación cómo hacer preguntas y reaccionar de una forma u otra según la respuesta sea correcta o no.

Analicemos el programa de la figura 24. Al presionar sobre la bandera verde, se nos hace una pregunta («¿En qué mes es la feria de Alba-

cete?») y el personaje cambia su postura hasta que le damos una respuesta. Con los condicionales (comandos si y si no), el objeto actúa de una manera u otra según le demos una respuesta correcta o no. Si la respuesta es septiembre habremos acertado, se oirá un aplauso, nuestro personaje cambiará de expresión mediante un cambio de disfraz y nos dirá «muy bien». Si damos cualquier respuesta que no sea septiembre, habremos fallado. El personaje adoptará una expresión diferente y nos dirá cuál es la respuesta correcta, a la vez se oirá un sonido de fallo. Finalmente, el personaje-objeto cambiará su expresión y nos invitará a visitar la feria. Al mismo tiempo se oirá una música de fiesta.

Si ejecutamos de nuevo las instrucciones el personaje tendrá la expresión con la que haya quedado en la ejecución anterior. Para que cambie y aparezca siempre con el disfraz inicial es necesario colocar al inicio del programa un comando que cambie el disfraz.

La figura 25 muestra el momento en el que el *personaje* del programa realiza la pregunta. El programa completo puede ejecutarse en la dirección <https://scratch.mit.edu/projects/173706311/>.

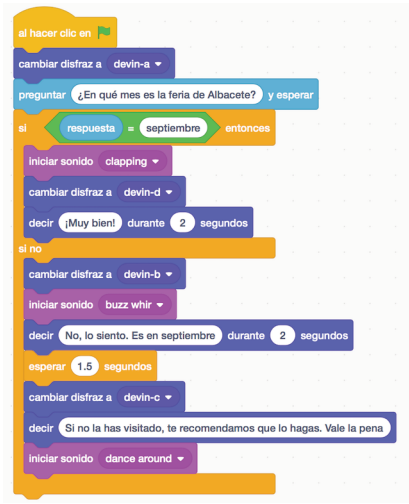

Figura 24. Comandos del programa de preguntas y respuestas

Controlando el movimiento

Vamos a dibujar un circuito por el que se ha de mover un objeto. Si el objeto choca con alguna de las paredes finaliza el juego. Para que el coche gire usaremos las teclas de control de nuestro ordenador.

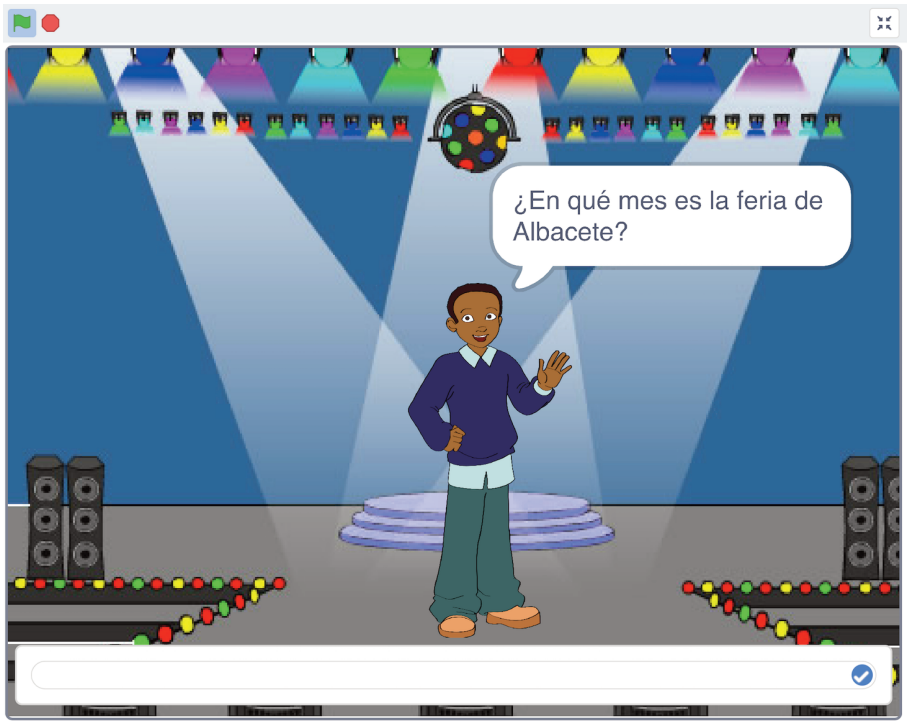

Figura 25. Resultado del programa de preguntas y respuestas

Lo primero que vamos a hacer es dibujar nuestro propio escenario yendo a **Dibujar fondo nuevo**. Dibujaremos un gran rectángulo de bordes negros y dentro de él otro rectángulo mucho más pequeño. Inicialmente colocaremos el objeto en la esquina superior izquierda y mirando hacia la derecha. Se detenga el objeto donde se detenga, haremos que, al presionar sobre la bandera verde, se sitúe en la posición inicial. El objeto se moverá en línea recta mientras no cambiemos su dirección.

Podemos dibujar muy diversos circuitos y hacer que el juego se complique a nuestro gusto. La figura 26 es una captura del escenario de Scractch para este ejemplo, en el que se puede ver el *personaje* seleccionado junto con el rectángulo dibujado.

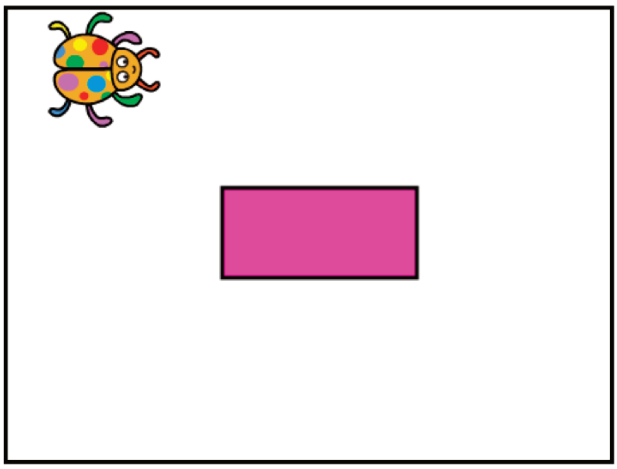

Figura 26. Escenario para el programa de control del movimiento

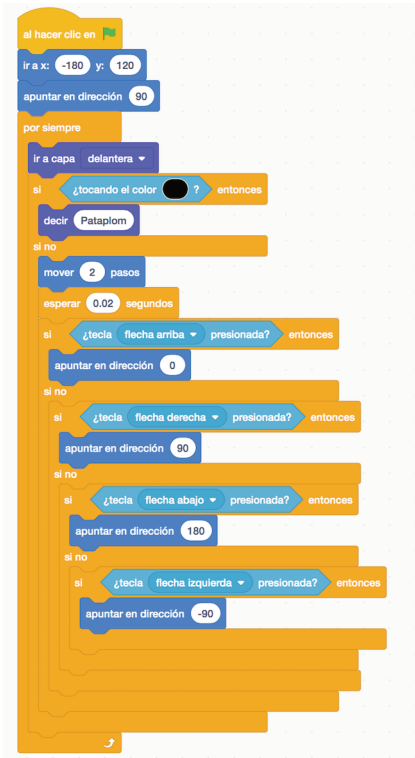

Figura 27. Programa del ejemplo de control del movimiento

Figura 28. Creación de una variable numérica con el menú de Datos para controlar la velocidad del *personaje*

El programa relacionado con este ejemplo se puede ver en la figura 27.

Se puede ver cómo evoluciona el objeto en <https://scratch.mit.edu/projects/173870675/>.

Más difícil todavía

Para finalizar, haremos algo similar a lo anterior, pero en este caso podremos aumentar la velocidad del objeto pulsando una tecla y también introduciremos un contador que haga que el vehículo se detenga al recorrer el circuito completo una sola vez, de manera que si presionamos la flecha derecha el objeto entenderá que ha recorrido el circuito completo y parará en el centro de la pista y nos dirá que ya hemos acabado el juego.

En este ejemplo se trabaja con variables que hemos de crear primero, desde la categoría **Datos** (ver figura 28). En este caso se trabajará con dos variables **contador**, destinado a contar las veces que pulsamos la flecha **hacia la derecha** y **velocidad** que nos permitirá aumentar la velocidad del objeto.

Para conseguir nuestro objetivo podríamos usar un programa como el de la figura 29. Se pueden ver los resultados de su ejecución en <https://scratch.mit.edu/projects/173876915/>.

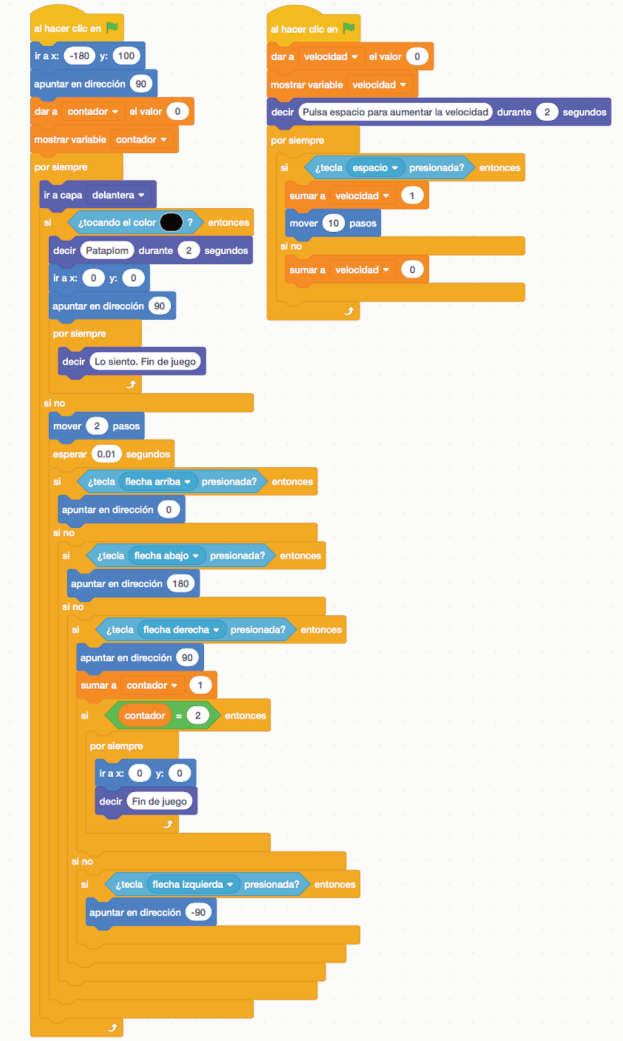

Figura 29. Programa para
controlar el movimiento usando las flechas
y aumentar la velocidad pulsando la barra espaciadora

Ventajas de usar Scratch

Una vez que hemos visto qué es Scratch y su uso para el desarrollo de programas, vamos a presentar las principales ventajas para trabajar con niños y adolescentes. Las hemos dividido en dos grupos: ventajas para el docente (es decir, que facilidades tiene el utilizar Scratch para enseñar) y ventajas para el alumno (es decir, qué capacidades adquieren los estudiantes con Scratch).

Ventajas para el docente:

— Es un programa gratuito y software libre, permitiendo a las instituciones educativas su uso sin limitaciones por el coste de licencias.
— Está disponible para los principales sistemas operativos (Windows, Mac, Linux) y, por tanto, facilita su implantación en el aula ya que el docente no tiene que comprobar el sistema operativo disponible.
— Se puede trabajar con Scratch on-line o instalándolo localmente en un ordenador y trabajar sin red, lo que permite una mayor flexibilidad en el aula.
— Se trata de un lenguaje de programación visual y que además incorpora elementos audiovisuales (vídeos, fotos, grabaciones, etc.), lo cual lo hace más atractivo para los alumnos.
— Los materiales elaborados se pueden insertar en una página web, permitiendo una mayor difusión del trabajo hecho y una futura reutilización de estos materiales.
— Se puede utilizar de forma transversal en una gran parte de las áreas de conocimiento.
— Finalmente, también se trata de una plataforma multilingüe, lo que permite trabajar contenidos en otros idiomas.

Ventajas para el alumno:

— Con Scratch se lleva a cabo un inicio en la programación de un modo visual y casi podríamos decir que jugando con piezas de un puzle.
— Se produce un desarrollo de la capacidad de resolución de problemas al desarrollar la capacidad de dividirlo en pequeños subproblemas más sencillos.
— Se produce un acercamiento a la programación desde dentro, facilitando la comprensión de conceptos lógicos y algoritmos matemáticos.
— Con la libre programación se fomenta la creatividad del alumno al poder construir programas sobre todo aquello que se le pueda ocurrir.
— Se desarrolla el pensamiento sistémico y de ordenación teniendo contacto directo con el modo en que funciona cada elemento, así como en la forma en que este se relaciona con otros elementos o pasos dentro del programa.

— Mejora la habilidad de comprensión de los alumnos mediante la interacción entre lo que se programa y los resultados instantáneos que se muestran.
— Debido a la interacción instantánea con la plataforma también se fomenta la capacidad de autocorrección, ya que, si no obtienen los resultados esperados, pueden cambiar fácilmente lo programado encajando y desencajando los bloques del programa con el ratón.
— La capacidad reflexiva antes de tomar una decisión también se fomenta con Scratch por el simple hecho de que se trata de encajar piezas de un puzle y los alumnos deben reflexionar sobre qué bloque tendrán que añadir en cada paso de la programación del algoritmo.
— Finalmente, Scratch puede ayudar a reforzar o entender conceptos de otras asignaturas o del propio colegio fomentando la relación entre materias y/o áreas de conocimiento.

Conclusiones

Scratch es un entorno sencillo para introducir la programación a los alumnos de últimos años de primaria, secundaria y bachillerato. Los programas operan sobre una serie de imágenes o *personajes* que actúan en un escenario, de manera que resulta muy visual y atractiva para el alumno. Dado que los comandos vienen representados por bloques y que los programas se construyen combinando unos bloques con otros, la creación de los mismos es muy sencilla.

Scratch dispone de una gama bastante amplia de comandos para manejar los *personajes*, de manera que se pueden crear desde programas sencillos que muevan de un lado a otro el *personaje*, hasta programas más complejos que permitan moverlos con el teclado e incluso interacciones entre varios *personajes*. A pesar de la sencillez y facilidad de uso, Scratch fuerza a los alumnos a ser cuidadosos y prestar atención a la hora de desarrollar el programa de un algoritmo, de manera que les obliga a trabajar paso a paso y con rigor en la resolución de problemas.

Además de los ejemplos presentados en este artículo, en la web de Scratch se pueden encontrar muchos otros recursos que pueden ser utilizados en el aula. Este uso podría ser de dos tipos. Por un lado, los alumnos podrían acceder a los recursos y examinar los programas creados por otras personas para aprender la programación con Scratch y, por otro lado, se puede pedir a los alumnos que intenten reproducir desde cero un programa ya creado y luego comparar su programa con el original. La primera opción puede ser mejor en las etapas iniciales del uso de Scratch para familiarizarse con él y ver todo su potencial, mientras que la segunda opción puede ser útil a la hora de proponer actividades a alumnos más avanzados en el uso de Scratch. Además, en la web de Scratch hay numerosos tutoriales con instrucciones para la construcción de programas paso a paso que se podrían utilizar.

Por último, pensamos que Scratch puede ser interesante para desarrollar actividades que abarquen otras materias. Por ejemplo, desarrollar programas para realizar preguntas sobre otras materias (como, por ejemplo, Lengua) de manera que se compruebe si la respuesta proporcionada es correcta o no. Para la asignatura de Biología, se podrían utilizar *personajes* que representen distintos tipos de animales dentro de un ecosistema y que interactúen entre ellos.

Bibliografía

Badger, M. (2014), *Scratch 2.0 Beginner's Guide*, 2nd edition, Packt Publishing, Birmingham.
Breen, D. (2015), *Scratch For Kids For Dummies,* For Dummies, New Jersey.
Dickins, R., L. Stowell y J. Melmoth (2015), *Coding for beginners using Scratc,* Usborne Publishing, London.
Hijón, R. (2018), *Piensa y programa con Scratch… en casa y en clase,* Anaya, Madrid.
Lakeman, L. (2014), S*tep by Step SCRATCH Programming*, Edco Publishing, Dublin.
Lifelong Kindergarten (2018), *Scratch*, <https://scratch.mit.edu>.
Marji, M. (2014), *Learn to program with Scratch*, No Starch Press, San Francisco.
McManus, S. (2013), *Scratch Programming in easy steps*, In Easy Steps Limited, Warwickshire.

Milonovich, B. (2013), *Scratch Cookbook*, Packt Publishing, Birmingham.

Sweigart, A. (2016, *Scratch Programming Playground*, No Starch Press, Toronto.

Vlieg, E. (2014), *Beyond Basic Scratch: Beyond Scratch programming basics,* Amazon Digital Services LLC.

— (2014), *Basic Scratch: An introduction to the Scratch programming language,* Amazon Digital Services LLC.

Woodcock, J. (2015), *Coding games in Scratch*, DK Children, New York.

Virgilio Gómez Rubio
Universidad de Castilla-La Mancha
<virgilio.gomez@uclm.es>

María José Haro Delicado
IES Al-Basit, Universidad de Castilla-La Mancha
<mariajose.haro@uclm.es>

Francisco Palmí Perales
Universidad de Castilla-La Mancha
<francisco.palmi@uclm.es>

* El presente artículo está escrito usando la versión 2.0 de Scratch, que era la última cuando se redactó. Sin embargo, el largo proceso de publicación en *Suma* (recepción, paso por *referees,* maquetación…), ha dado lugar a que en el momento de imprimirse, ya se encuentre disponible la versión 3.0. Ese es el motivo de que algunas de las pantallas y menús que aparecen no coinciden exactamente con sus equivalentes en la versión actual. Los autores no son responsables de esta *desactualización.*