J. Ángel Velázquez Iturbide

Reglas de Comportamiento del Lenguaje Basado en Bloques ScratchJr

Número 2024-01

Serie de Informes Técnicos DLSI1-URJC
ISSN 1988-8074
Departamento de Informática y Estadística
Universidad Rey Juan Carlos

Índice

Reglas de Comportamiento del Lenguaje Basado en Bloques ScratchJr

J. Ángel Velázquez Iturbide

Departamento de Informática y Estadística, Universidad Rey Juan Carlos,
C/ Tulipán s/n, 28933, Móstoles, Madrid
angel.velazquez@urjc.es

Resumen. Se presenta una descripción de la dinámica del lenguaje basado en bloques ScratchJr en forma de reglas de comportamiento. El conjunto de reglas presentado es el resultado de una indagación previa y han sido evaluadas con profesores. Las reglas de comportamiento están dirigidas principalmente a formadores de profesores de Educación Infantil, aunque también pueden resultar útiles a profesores de Infantil que quieran conocer el lenguaje de forma sistemática y no meramente intuitiva.

Palabras clave: Programación basada en bloques, ScratchJr, reglas de comportamiento, formación de profesores.

1 Introducción

ScratchJr es un lenguaje de programación basado en bloques concebido para niños de 5 a 7 años [1]. Pueden encontrarse descripciones informales de la dinámica del lenguaje, normalmente mediante ejemplos [2][3][4][5], pero no existe una definición detallada del lenguaje. Sin embargo, es necesario que los formadores en ScratchJr de los profesores de Educación Infantil dispongan de una descripción precisa y detallada. Estos formadores pueden ser profesores con cierta formación informática, o al menos experiencia. Hay varias razones por las que estos formadores deben disponer de dicha descripción. Por un lado, sus alumnos (profesores de Educación Infantil actuales o futuros) serán adultos, por lo que es adecuado que reciban una formación basada en la comprensión más que en la mera intuición. Incluso algunos de estos alumnos podrían tener interés en detalles del funcionamiento de ScratchJr que exijan un conocimiento preciso. Además, su conocimiento del lenguaje debe ser correcto, a riesgo de que los profesores de Infantil aprendan (y posteriormente enseñen) ScratchJr con errores. Por último, la comprensión de la dinámica del lenguaje les permitirá comprender comportamientos inesperados de programas ScratchJr.

En este informe describimos el lenguaje de programación ScratchJr mediante reglas de comportamiento [6]. Se han obtenido como resultado de varias actividades: una indagación estructurada [7], un análisis de la implementación del lenguaje[1] e incluso algunos detalles mediante la lectura de publicaciones de los autores del

[1] https://github.com/jfo8000/ScratchJr-Desktop/

lenguaje [1][4]. Posteriormente, fueron evaluadas por alumnos de un máster para profesorado en el que aprenden ScratchJr y por profesores universitarios de grado en el que enseñan ScratchJr. La evaluación se encuentra descrita con todo detalle en otro informe técnico [8].

Las reglas de comportamiento se presentan en el Apéndice. Se estructuran en cinco conjuntos: programas y bloques, escenario y movimientos, apariencia y sonido, paralelismo y eventos, y control y terminadores. En sentido estricto, el primer conjunto no contiene reglas de comportamiento, sino una descripción parcial del entorno de programación y una descripción de la sintaxis del lenguaje que sirven de marco de referencia de los alumnos para las propias reglas.

Agradecimientos. Este trabajo se ha financiado con los proyectos de investigación PROGRAMA de la Universidad Rey Juan Carlos (ref. M3035) y del Ministerio de Ciencia e Innovación (ref. PID2022-137849OB-I00).

Referencias

1. Flannery, L.P., Kazakoff, E.R., Bontá, P., Silverman, B., Bers, M.U., Sullivan, A.: Designing ScratchJr: Support for early childhood learning through computer programming. En: Proceedings of the 12th International Conference on Interaction Design and Children (IDC'13). ACM DL, 2013, págs. 1-10, DOI 10.1145/2485760.2485785
2. ScratchJr – Aprende – Guía de bloques. https://www.scratchjr.org/pdfs/block-descriptions.pdf
3. ScratchJr – Enseña – Actividades. https://www.scratchjr.org/teach/activities
4. Bers, M.U., Resnick, M.: The Official ScratchJr Book: Help Your Kids Learn to Code. No Start Press (2016)
5. Bers, M.U., Sullivan, A.: ScratchJr Coding Cards: Creative Coding Activities. No Start Press (2018)
6. Durán, R., Sorva, J, Seppälä,, O.: Rules of program behavior. ACM Transactions on Computing Education 21, 4 (noviembre 2021), artículo 33. DOI 10.1145/3469128
7. Velázquez Iturbide, J.Á.: Una indagación del comportamiento del lenguaje ScratchJr. En: Serie de Informes Técnicos DLSI1-URJC, no. 2021-03, Departamento de Lenguajes y Sistemas Informáticos I, Universidad Rey Juan Carlos, 2021.
8. Velázquez Iturbide, J.Á.: Evaluación de patrones elementales y reglas de comportamiento de ScratchJr. En: Serie de Informes Técnicos DLSI1-URJC, no. 2024-03, Departamento de Lenguajes y Sistemas Informáticos I, Universidad Rey Juan Carlos, 2021.

Apéndice: Reglas de Comportamiento de ScratchJr

Se presenta en este documento una descripción del comportamiento del lenguaje de programación ScratchJr en forma de afirmaciones o reglas. Para facilitar su lectura y comprensión, se han agrupado en cinco conjuntos de reglas, que pueden leerse sucesivamente. Dentro de cada conjunto de reglas, los bloques suelen describirse en el mismo orden en que aparecen en el entorno de ScratchJr. Cada conjunto de reglas termina con varios ejemplos de código que pretenden ejemplificar algunas de las reglas del conjunto.

Cada conjunto de reglas describe una o varias categorías de bloques:

1. Programas y bloques. Este conjunto de reglas describe la sintaxis de los bloques y (parcialmente) el entorno de programación. Se ha incluido por completitud, pero el conjunto puede omitirse si se está interesado únicamente en el comportamiento de los bloques.
2. Escenario y movimientos.
3. Apariencia y sonido. Se incluye también el bloque “Fijar velocidad” porque afecta a los bloques de movimiento o de apariencia.
4. Paralelismo y eventos.
5. Control y terminadores.

Conjunto de reglas #1 – Programas y bloques

§1 ScratchJr es un lenguaje de programación basado en bloques ideado para niños.

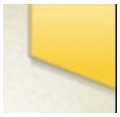
§1.1 Un programa ScratchJr se llama proyecto.

§1.2 El icono de casa del entorno de programación de ScratchJr muestra los proyectos creados. El usuario puede seleccionar uno de estos proyectos o crear otro nuevo.

§1.3 La operación de crear algo nuevo suele representarse en el entorno de programación de ScratchJr con el icono del signo más '+'.

§1.4 El entorno de programación de ScratchJr permite crear y ejecutar un proyecto en la misma pantalla, que está dividida en distintas áreas. La mayoría de las áreas no se describen en este documento.

§1.5 Puede darse un nombre al proyecto pulsando en el icono de información (en la esquina superior derecha, de color amarillo). La longitud máxima admitida como nombre del proyecto es 30 caracteres.



§1.6 Si se quiere guardar un proyecto desarrollado, debe pulsarse el icono casa, tras lo cual se vuelve a la lista de proyectos disponibles.

§2 Un proyecto está formado por una o varias páginas, cada una con uno o varios personajes y sus secuencias de bloques más un fondo mostrado en el escenario.

§2.1 Por defecto, un proyecto contiene una sola página con el personaje Gato y un fondo blanco en el escenario.

§2.2 Las páginas se estudian en la categoría de bloques terminadores.

§2.3 Los personajes de una página aparecen en la parte izquierda de la pantalla, mediante su nombre y una miniatura de su imagen.

§2.4 Un personaje se puede añadir al programa (pulsando el icono '+'), borrar del mismo (pulsando el personaje durante un tiempo) o incluso cambiar su nombre o su imagen (pulsando en el icono de pincel).

§2.5 Un personaje puede añadirse varias veces. También pueden suprimirse personajes añadidos previamente.

§2.6 Un personaje puede tener cero o más secuencias de bloques. Si hay varias copias de un mismo personaje, cada copia tiene su propio programa.

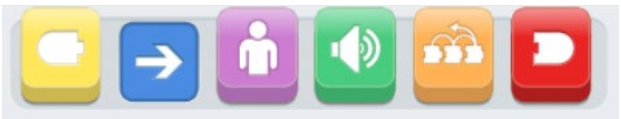
§2.7 Para crear el programa de un personaje, se selecciona primero el personaje y después se arrastran bloques desde las áreas de categorías y paleta de bloques a la zona de programación, ensamblándolos entre sí.

§2.8 Puede copiarse cualquier secuencia de bloques de un personaje a otro arrastrándolo desde la zona de programación del primero al segundo.

§3 Un proyecto de ScratchJr se construye con un conjunto restringido de bloques.

§3.1 Una secuencia de bloques se llama guion y se lee de izquierda a derecha.

§3.2 ScratchJr proporciona 28 bloques.

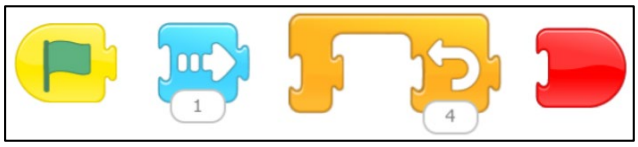


§3.3 Los bloques de ScratchJr se agrupan en 6 categorías, cada una reconocible por un color:

a. Bloques disparadores, de color amarillo.
b. Bloques de movimiento, de color azul.
c. Bloques de apariencia, de color lila.
d. Bloques de sonido, de color verde.
e. Bloques de control, de color naranja.
f. Bloques terminadores, de color rojo.

§3.4 Los bloques de movimiento y de apariencia producen efectos visibles en el escenario, y los de sonido reproducen sonidos. Sin embargo, los demás bloques controlan el orden o velocidad de ejecución del programa.

§3.5 Hay 4 formas de bloque, que determinan con cuántos y qué bloques puede ensamblarse. La mayor parte de los bloques pueden tener un bloque a su izquierda y otro a su derecha.

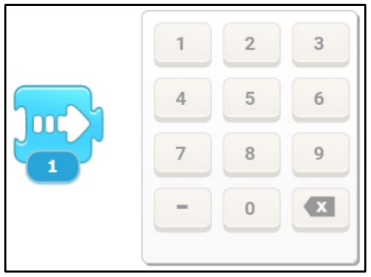


§3.6 Los bloques disparadores sólo pueden ensamblarse con un bloque a su derecha, es decir, sólo pueden ir al comienzo (a la izquierda) de un guion.

§3.7 Los bloques terminadores sólo pueden ensamblarse con un bloque a su izquierda, es decir, sólo pueden ir al final (a la derecha) de un guion.

§3.8 Un bloque “Repetir” puede ensamblarse por su izquierda y su derecha y, además, puede englobar a una secuencia de bloques.

§3.9 Algunos bloques tienen un parámetro, puede particularizarse. Puede ser un número, un texto o una opción de menú a seleccionar. Sea un valor o un menú a desplegar, siempre se muestra en la parte inferior del bloque.



§3.10 El parámetro numérico de un bloque admite valores positivos y negativos de dos dígitos (es decir, comprendidos entre -99 y 99), salvo los bloques de control “Esperar” y “Repetir”, que no admiten valores negativos.

§4 La ejecución de un proyecto consiste en que se realicen las acciones indicadas por los guiones de los personajes.

§4.1 Cualquier bloque o secuencia de bloques situado en la zona de programación puede ejecutarse pulsándolo con el ratón.

§4.2 La forma habitual de ejecutar un proyecto es pulsar el icono de bandera verde.

§4.3 Tras pulsar el icono de bandera verde, éste es sustituido por un icono de hexágono rojo. La pulsación del icono de hexágono rojo produce la finalización de la ejecución del programa.

§4.4 La ejecución de una secuencia de bloques consiste en la ejecución uno a uno, de izquierda a derecha, de dichos bloques. La ejecución de un bloque comienza sólo tras terminar la ejecución del bloque anterior.

§4.5 La ejecución de una secuencia de bloques termina cuando termina la ejecución del último bloque.

§4.6 Existe la posibilidad de alterar la ejecución no secuencial de bloques o de que se inicie la ejecución de un guion sin haber pulsado la bandera verde. Dichas posibilidades se ven con las categorías de bloques disparadores, de control y terminadores.

Conjunto de reglas #2 – Escenario y movimientos

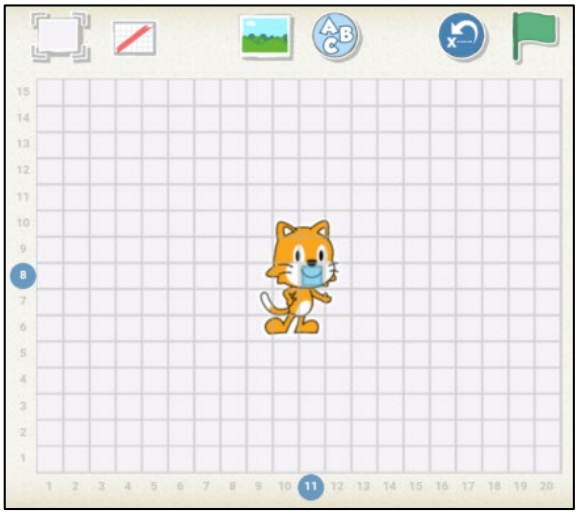
§5 El escenario es una malla sobre la cual los personajes realizan acciones.

§5.1 El escenario es blanco por defecto, pero puede elegirse un fondo de los proporcionados por el entorno e incluso crear fondos nuevos.

§5.2 El escenario consta de 20 celdas de ancho por 15 de alto.

§5.3 Cuando se añade un personaje, se coloca en el centro del escenario, es decir, en la celda 11ª contando desde la izquierda y la 8ª desde abajo.

§5.4 La malla puede hacerse visible pulsando en el icono "Cuadrícula" situado encima del escenario, como puede verse en la figura. El icono aparece tachado (segundo comenzando por la izquierda), indicando que se puede volver a pulsar para ocultar la malla. El personaje Gato está en su posición inicial.



§6 Los bloques de movimiento producen un cambio en la posición del personaje en el escenario. El cambio puede consistir en un avance, giro, salto o colocación directa. Para los bloques de avance, giro o salto:

§6.1 Un bloque de movimiento especifica el número de pasos, giros o la altura del salto que dará el personaje sobre el escenario.

§6.2 Un número cero no produce ningún cambio en la posición del personaje.

§6.3 El tiempo utilizado en ejecutar un bloque de movimiento es proporcional al número de pasos, giros o la altura del salto a realizar.

§7 Hay cuatro bloques distintos para avanzar en cada una de las direcciones del plano:

a. Bloque "Mover a la izquierda" 

b. "Mover a la derecha" 

c. "Subir" 

d. "Bajar" 

§7.1 La unidad de avance es una celda de la malla del escenario.

§7.2 Un número positivo de pasos produce que el personaje se oriente en la dirección del movimiento y avance dicho número de pasos en dicha dirección (véase el ejemplo 1). Un número negativo de pasos produce que el personaje avance dicho número de pasos en dirección contraria a la del bloque pero sin cambiar de orientación (véase el ejemplo 1).

§7.3 El escenario es circular en ambas dimensiones. Es decir, un personaje que sale por la derecha continúa avanzando por la izquierda y si sale por arriba continúa avanzando por abajo, y viceversa (véase el ejemplo 2).

§8 Hay dos bloques distintos para girar sobre el plano en cada sentido:

a. "Girar a la derecha" (en sentido horario) 

b. "Girar a la izquierda" (en sentido antihorario) 

§8.1 La unidad de giro son 30 grados.

§8.2 Un número positivo produce un número de giros en el sentido que indica el bloque (véase el ejemplo 3). Un número negativo produce un número de giros en sentido contrario al que indica el bloque (véase el ejemplo 3).

§9 El bloque "Saltar" hace que el personaje suba primero y baje después un cierto número de celdas, simulando un salto. 

§9.1 Si el salto es de un número negativo de pasos, primero baja y después sube.

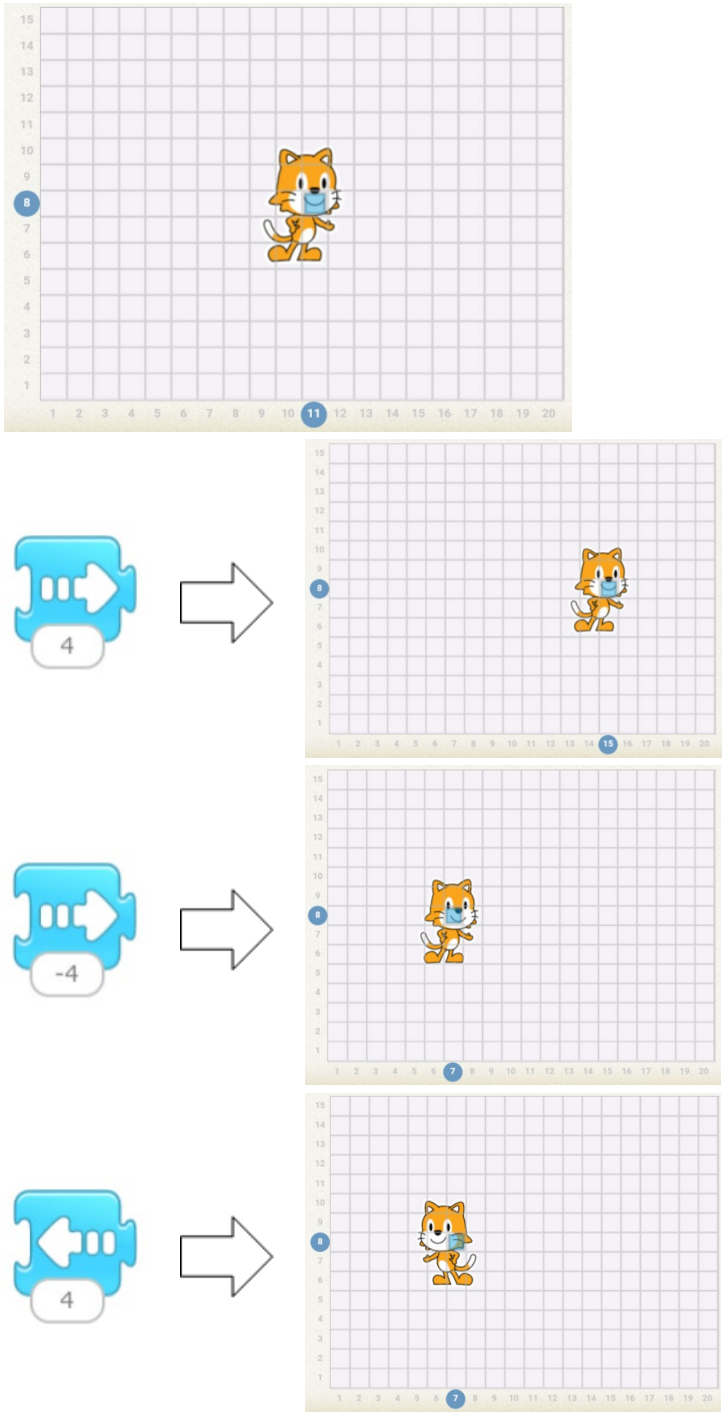
§9.2 El escenario no se comporta de forma circular con el bloque "Saltar". Con un número positivo, si al subir llega al borde superior (de altura 15), no avanza más. Después baja tantas casillas como indica su número (sin sobrepasar el borde inferior). (Véase el ejemplo 4.) Con un número negativo, si al bajar llega al borde inferior, tampoco avanza más. Después sube tantas casillas como indica su número (sin sobrepasar el borde superior). (Véase el ejemplo 5.)

§10 El bloque "Ir al inicio" coloca al personaje en la posición inicial que ocupaba antes de ejecutar el programa. 

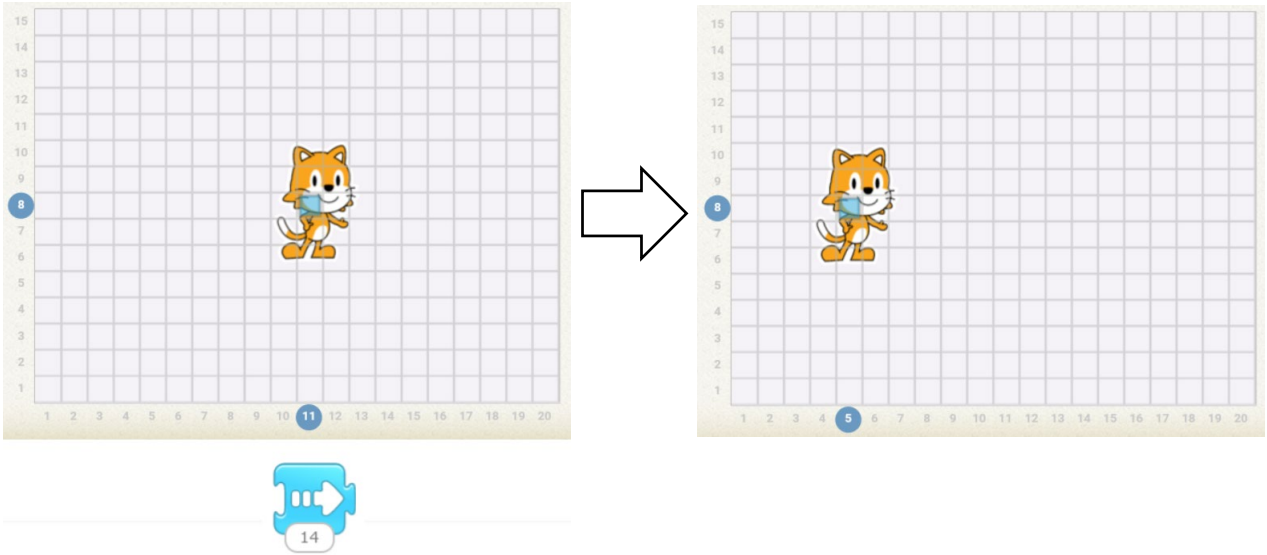
§10.1 Puede cambiarse la posición inicial de un personaje arrastrándolo a la posición deseada.

Ejemplos:

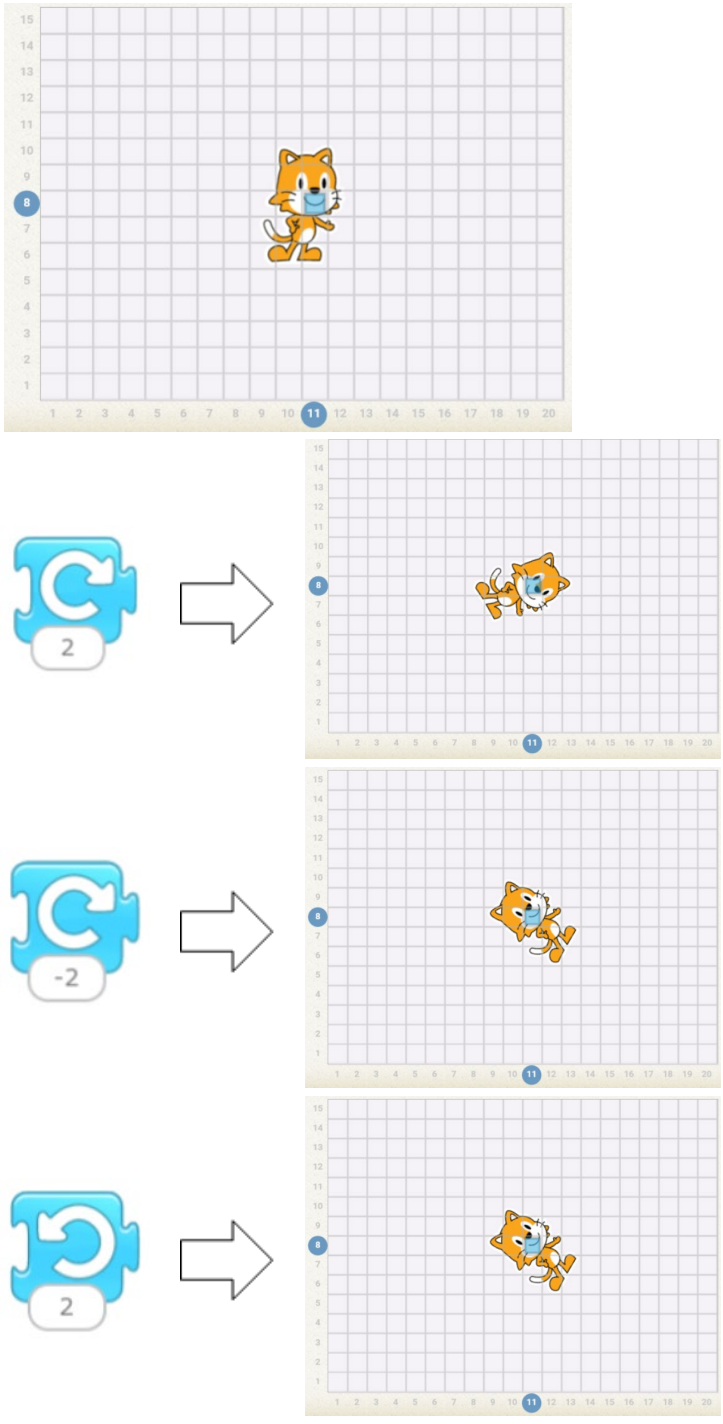
Ejemplo 1. Movimiento de un número de pasos.



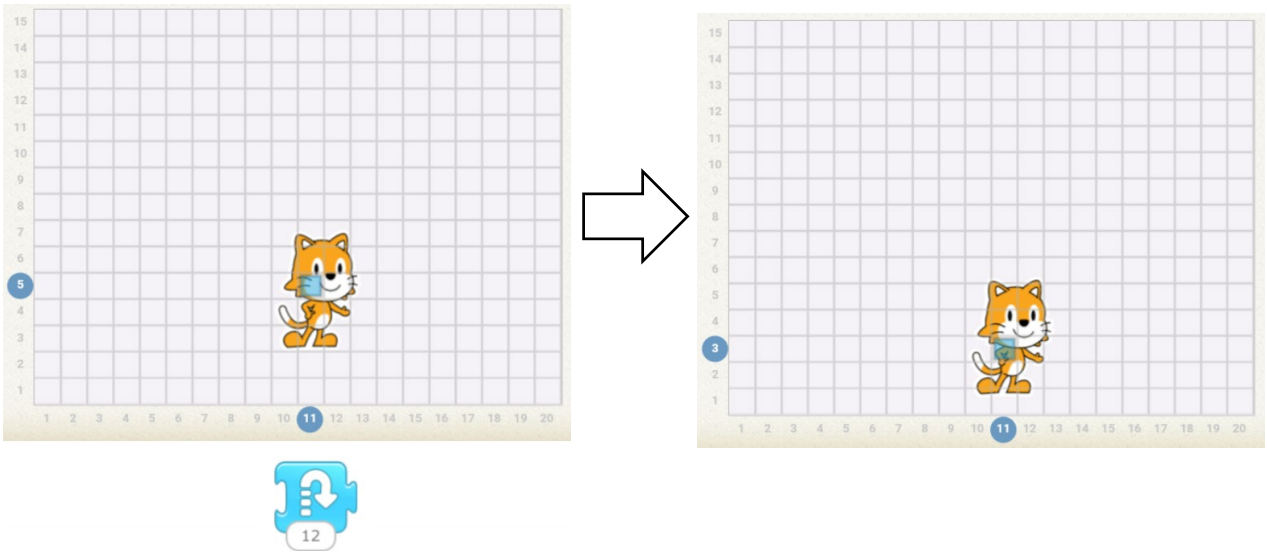
Ejemplo 2. Movimiento de un número de pasos que hace que el personaje aparezca por el lado contrario del escenario. Primero avanza 9 pasos hasta llegar al borde derecho del escenario; después, avanza 5 pasos comenzando desde el borde izquierdo.



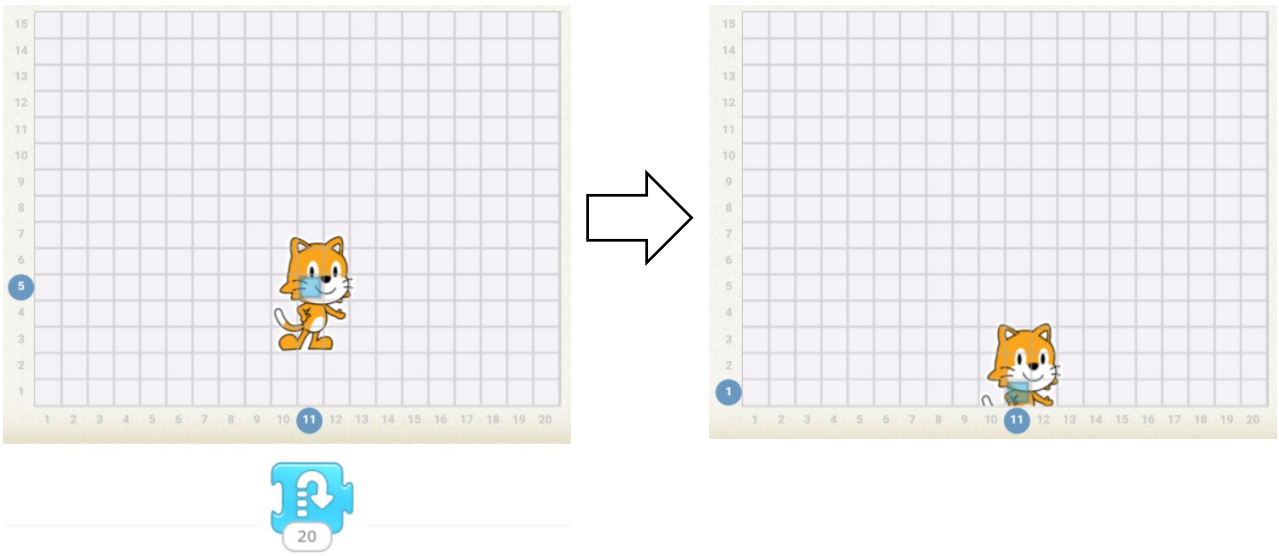
Ejemplo 3. Movimiento de un número de giros.



Ejemplo 4. Desde una altura 5, el gato sube 10 pasos (llegando al borde superior) y después baja 12, terminando a una altura 3.



Ejemplo 5. Desde una altura 5, el gato sólo sube 10 pasos hasta una altura 15 (borde superior) y después sólo baja 15 pasos hasta una altura 1 (borde inferior).



Conjunto de reglas #3 – Apariencia y sonido

§11 Los bloques de apariencia producen un cambio en el aspecto del personaje. El cambio puede consistir en un cambio de tamaño, que sea o no visible o que esté o no diciendo algo.

§12 El bloque "Decir" muestra un mensaje en forma de bocadillo de comic, como si lo dijera el personaje (véase el ejemplo 6).

§12.1 La longitud máxima del texto admisible en un bloque "Decir" es de 50 caracteres.

§12.2 El texto se muestra durante un tiempo proporcional al número de caracteres, aproximadamente durante un segundo por cada 7 caracteres.

§13 Hay tres bloques distintos para cambiar el tamaño del personaje:

a. Bloque "Crecer" 

b. Bloque "Disminuir" 

c. Bloque "Restablecer tamaño" 

§13.1 Cada personaje tiene un tamaño inicial distinto.

§13.2 Los bloques "Crecer" y "Disminuir" producen un cambio proporcional del tamaño del personaje, dependiendo del factor indicado en el bloque. De forma orientativa, un factor 10 en los bloques "Crecer" y "Disminuir" produce un tamaño doble (véase el ejemplo 6) o mitad, respectivamente.

§13.3 Cada personaje tiene un tamaño máximo y un tamaño mínimo que puede alcanzar. Una vez alcanzado el tamaño máximo, ya no produce efecto el bloque "Crecer". Lo mismo sucede con el tamaño mínimo y el bloque "Disminuir" (véase el ejemplo 7).

§13.4 El cambio de tamaño se percibe visualmente de forma gradual, tardando un tiempo proporcional al cambio de tamaño a realizar.

§13.5 El bloque "Restaurar tamaño" restablece el tamaño que el personaje tiene por defecto.

§14 Hay dos bloques para mostrar u ocultar a un personaje:

a. Bloque "Mostrar" 

b. Bloque "Ocultar" 

§14.1 El efecto de ambos bloques se produce gradualmente desde su estado inicial hasta ser totalmente visible o invisible.

§15 El bloque "Fijar velocidad" permite fijar la velocidad a la que se ejecutan los bloques de movimiento y de apariencia (salvo "Decir") de un personaje (véase el ejemplo 8).

§15.1 Puede tomar tres valores, que se eligen mediante un menú desplegable en el propio bloque:

a. Velocidad normal. Es el valor por defecto. Es la mitad de la velocidad rápida.
b. Velocidad rápida.
c. Velocidad lenta. Es un tercio de la velocidad rápida.

§15.2 El bloque afecta a todo el programa del personaje.

§16 Cada personaje tiene un estado, cuyo valor varía durante la ejecución del programa.

§16.1 El estado de un personaje está formado por los siguientes atributos:

a. Posición (celda del escenario) en el eje horizontal. Varía entre 1 (extremo izquierdo) y 20 (extremo derecho).
b. Posición (celda del escenario) en el eje vertical. Varía entre 1 (extremo inferior) y 15 (extremo superior).
c. Ángulo. Puede estar en su orientación angular original o haber realizado giros múltiplos de 30º.
d. Orientación. Puede estar orientado hacia la izquierda o la derecha.
e. Tamaño. Varía entre un tamaño mínimo y un tamaño máximo específico de cada personaje.
f. Visibilidad. Un personaje puede estar visible u oculto.
g. Hablando. Un personaje puede estar diciendo algo o no.
h. Velocidad de ejecución. Un personaje puede ejecutar sus bloques de movimiento y apariencia (salvo "Decir") en velocidad normal, rápida o lenta.

§16.2 Al comenzar la ejecución de un programa, el personaje se hace visible y se restauran su posición inicial, ángulo, orientación y tamaño.

§16.3 Los valores iniciales de las posiciones horizontal y vertical de un personaje pueden cambiarse moviéndolo directamente sobre el escenario a la posición deseada.

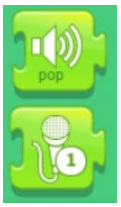
§16.4 Los atributos de tamaño, visibilidad y velocidad pueden cambiarse al valor deseado mediante la ejecución en la zona de programación del bloque correspondiente.

§17 Los bloques de sonido permiten reproducir un sonido.

§17.1 Hay dos bloques de sonido:

a. Bloque "Pop"

b. Bloque "Reproducir sonido grabado"



§17.2 Por defecto, el bloque "Reproducir sonido grabado" no puede usarse en un programa, lo cual se indica con un icono punteado de micrófono.

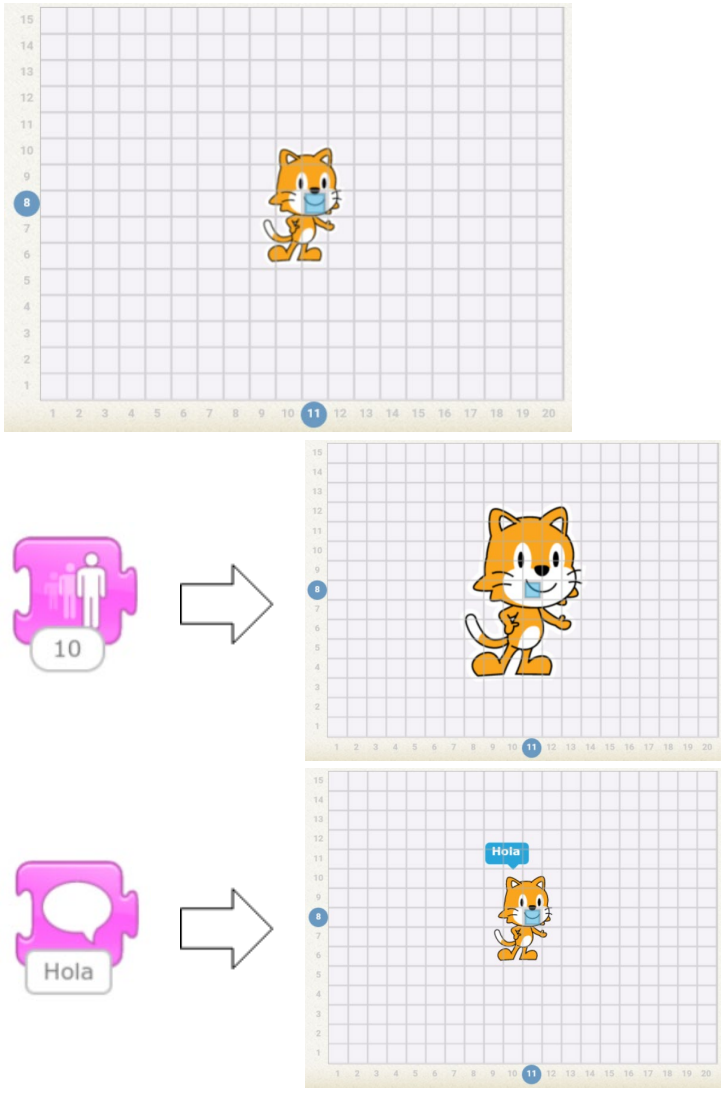
§17.3 Se puede pulsar en el icono punteado para grabar el sonido, frase o melodía que se desee, tras lo cual aparece el bloque correspondiente, etiquetado con un número de sonido.

§17.4 Cada sonido grabado se asocia solamente al personaje activo en ese momento.

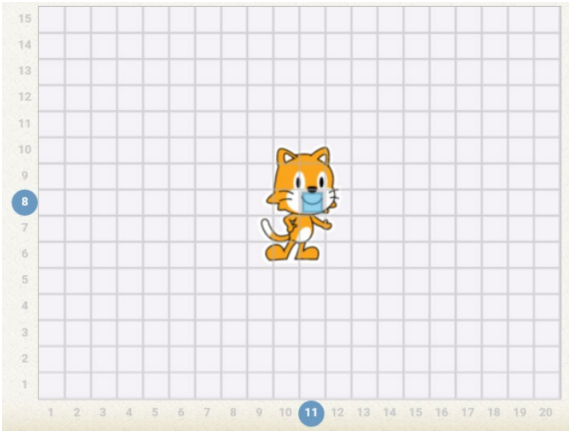
§17.5 Se pueden grabar hasta 5 sonidos por personaje.

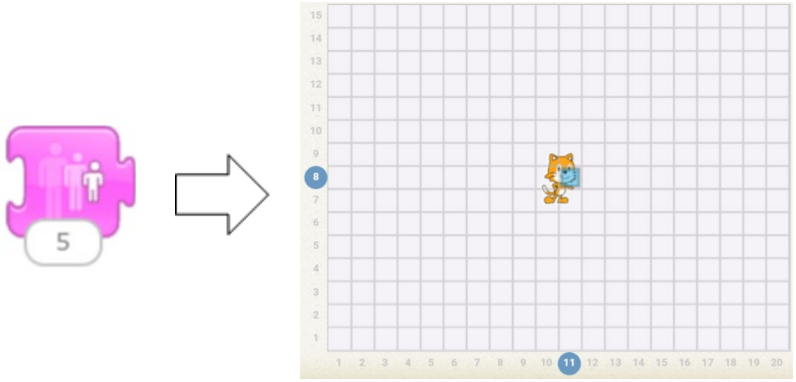
Ejemplos:

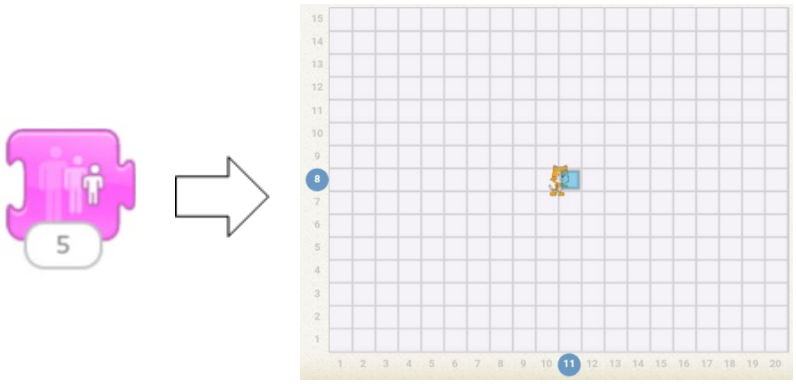
Ejemplo 6. Nuevo estado del gato tras ejecutar cada una de las siguientes operaciones, siempre a partir del siguiente estado inicial del gato en el escenario: tras aumentar su tamaño en 10 y mientras dice “Hola”.

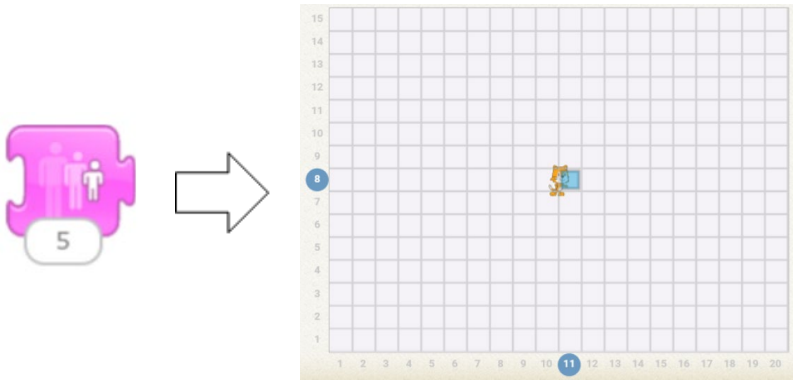


Ejemplo 7. Se muestra al gato en su tamaño inicial y tras ejecutar sucesivamente el bloque “Disminuir” en un factor de 5. Llega un momento en que ya no disminuye más.

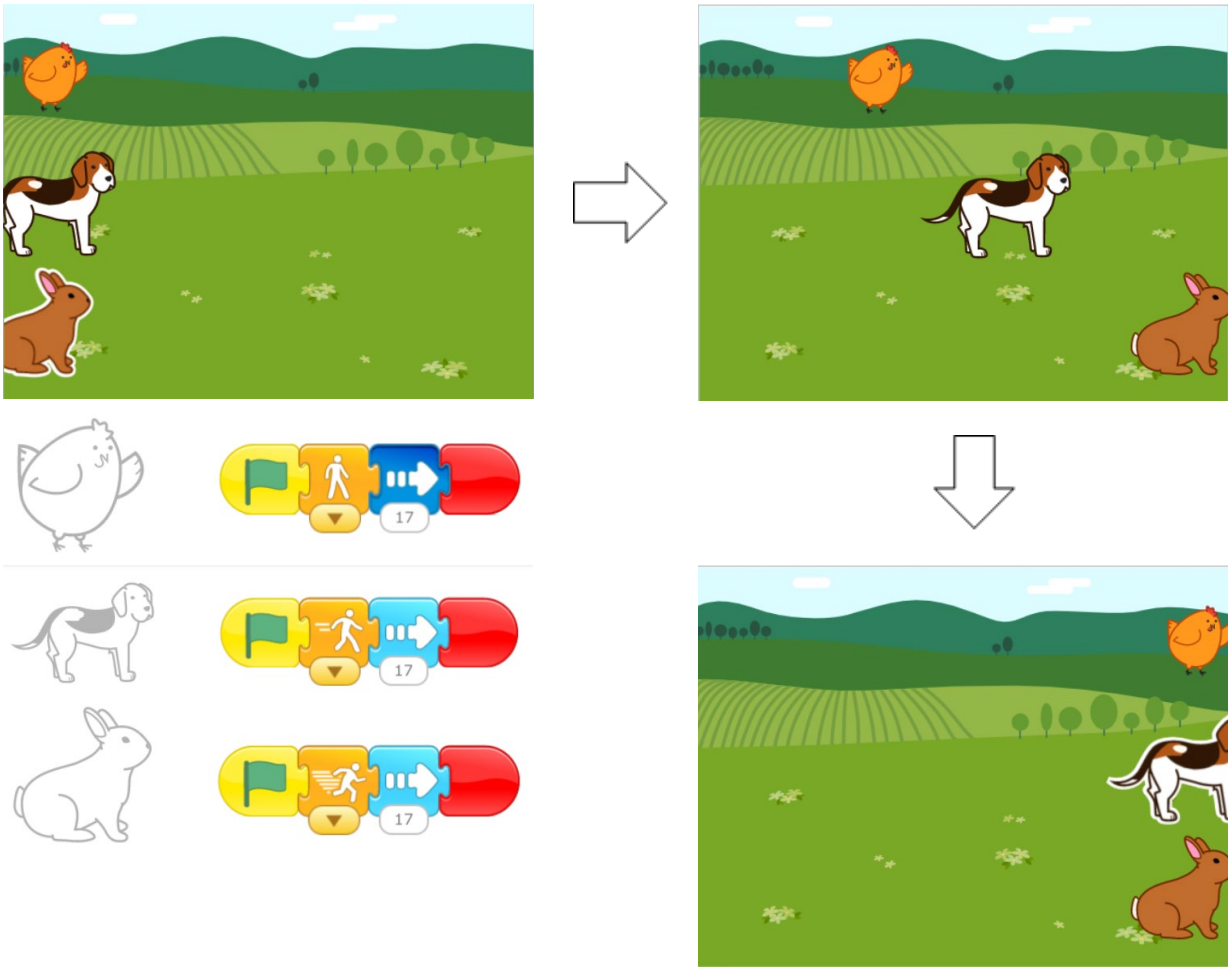








Ejemplo 8. Se muestra el escenario antes, durante y tras la ejecución del bloque “Mover a la derecha” de cada personaje.



Conjunto de reglas #4 – Paralelismo y eventos

§18 Un programa de ScratchJr puede presentar paralelismo, es decir, que varios guiones se ejecuten simultáneamente.

§18.1 El paralelismo de ScratchJr puede darse por dos circunstancias no excluyentes:

a. Un personaje tiene un programa formado por varios guiones (véase el ejemplo 23).
b. Existen varios personajes, cada uno con su propio programa (véase el ejemplo 8).

§18.2 Al presionar el icono de bandera verde, inician su ejecución simultánea todos los guiones encabezados a su izquierda por el bloque “Al pulsar bandera verde” (véase el ejemplo 8).

§18.3 De forma orientativa, podemos imaginar que la ejecución de varios guiones en paralelo se realiza ejecutando un bloque de cada guion, de forma alterna. Sin embargo, no podemos predecir el orden exacto de ejecución de sus bloques.

§19 Los bloques disparadores activan la ejecución de un guion cuando detectan que ha sucedido un evento.

§19.1 Hay cuatro bloques disparadores, cada uno capaz de detectar un evento distinto:

a. Bloque “Al presionar bandera verde” 

c. Bloque “Comenzar al tocar” 

d. Bloque “Comenzar al pulsar” 

b. Bloque “Comenzar con mensaje” 

§19.2 Un quinto bloque disparador (“Enviar mensaje”) produce un evento de envío de mensaje. 

§19.3 Cuando sucede un evento, lo detectan todos los bloques disparadores existentes de su tipo. Cada uno de estos bloques disparadores inicia la ejecución del guion que encabeza.

§20 El bloque “Al presionar bandera verde” detecta el evento producido por el usuario al pulsar el icono de bandera verde.

§20.1 El evento hace que todos los personajes vuelvan a su posición, ángulo, orientación, visibilidad y tamaño iniciales.

§20.2 A continuación, el evento activa todos los guiones de la página que comiencen por este bloque, pasando a ejecutarse en paralelo.

§21 El bloque “Comenzar al tocar” detecta el evento producido cuando dos personajes se tocan, es decir, cuando se solapan visualmente en el escenario (véase el ejemplo 9).

§21.1 El evento sólo activa en la página los guiones que comiencen por este bloque de los dos personajes que se tocan, pasando a ejecutarse en paralelo.

§21.2 Este bloque puede producir como efecto indeseado que el programa inicie o no termine su ejecución, incluso contra la voluntad del programador. En efecto, cuando los dos personajes se encuentran tocándose, se produce el evento, activándose los guiones de estos personajes que empiezan por el bloque “Comenzar al tocar”. La generación y captura

del evento puede producirse repetidamente (véanse los ejemplos 10 y 11). El usuario puede pararlos terminando la ejecución del programa (pulsando el icono de hexágono rojo).

§22 El bloque “Comenzar al pulsar” detecta el evento producido cuando el usuario pulsa sobre un personaje (véase el ejemplo 16).

§22.1 El evento sólo activa los guiones en la página del personaje tocado que comiencen por este bloque.

§23 Los dos bloques “Enviar mensaje” y “Comenzar con mensaje” se utilizan de forma coordinada para sincronizar operaciones o personajes.

§23.1 Pueden enviarse hasta 6 mensajes distintos, cada uno con un color diferente.

§23.2 La ejecución de un bloque “Enviar mensaje” con un color concreto produce el evento correspondiente. Como consecuencia, se activan todos los guiones de la página que comiencen por el bloque “Comenzar con mensaje” con el mismo color.

§23.3 Cuando un guion ejecuta un bloque de envío de mensaje, su ejecución queda provisionalmente en suspenso (véase el ejemplo 12).

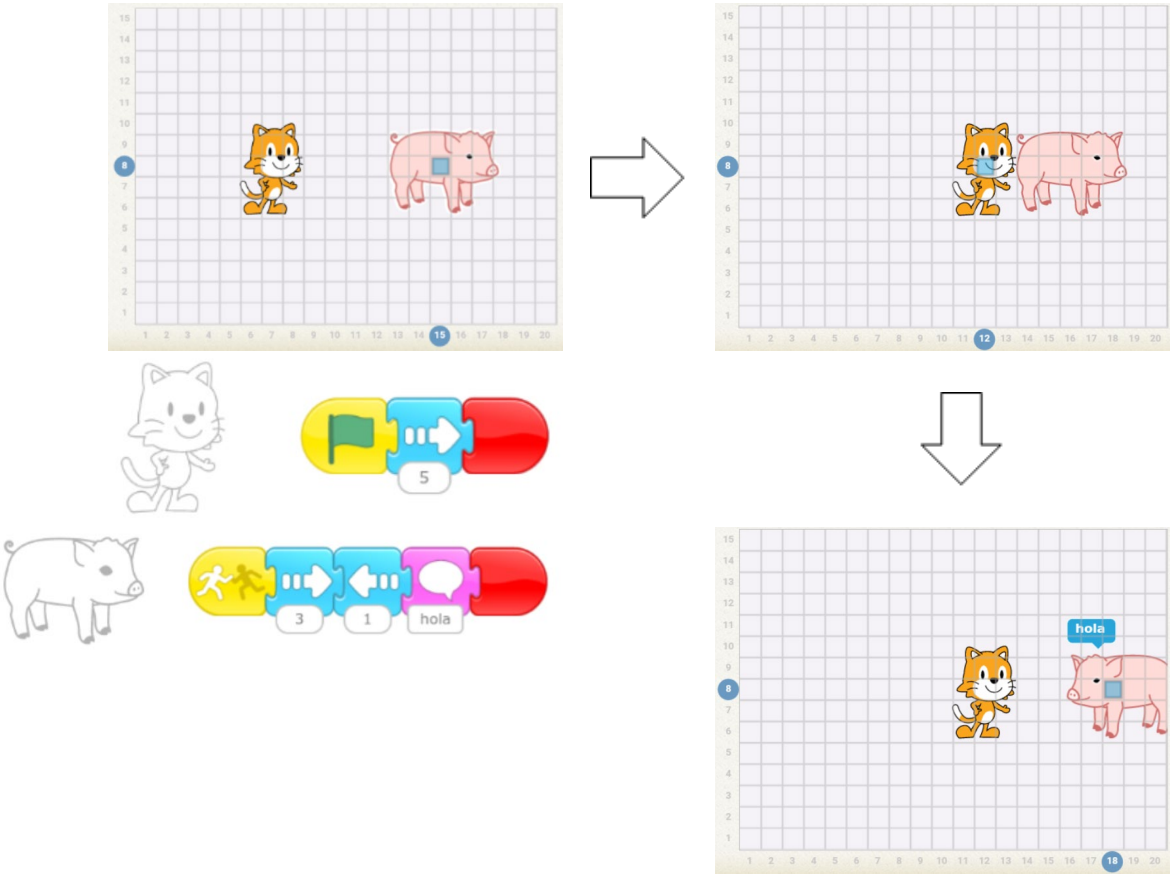
§23.4 Si no existe ningún guion que reciba un mensaje de un color dado, el guion que envió el mensaje reanuda su ejecución interrumpida.

§23.5 Si existen uno o varios guiones que reciben un mensaje de un color dado, el guion que envió el mensaje reanudará su ejecución interrumpida cuando terminen su ejecución todos los guiones activados (véase el ejemplo 13). Si algún guion activado envía otro mensaje, también se interrumpe su ejecución provisionalmente y así sucesivamente (véase el ejemplo 14).

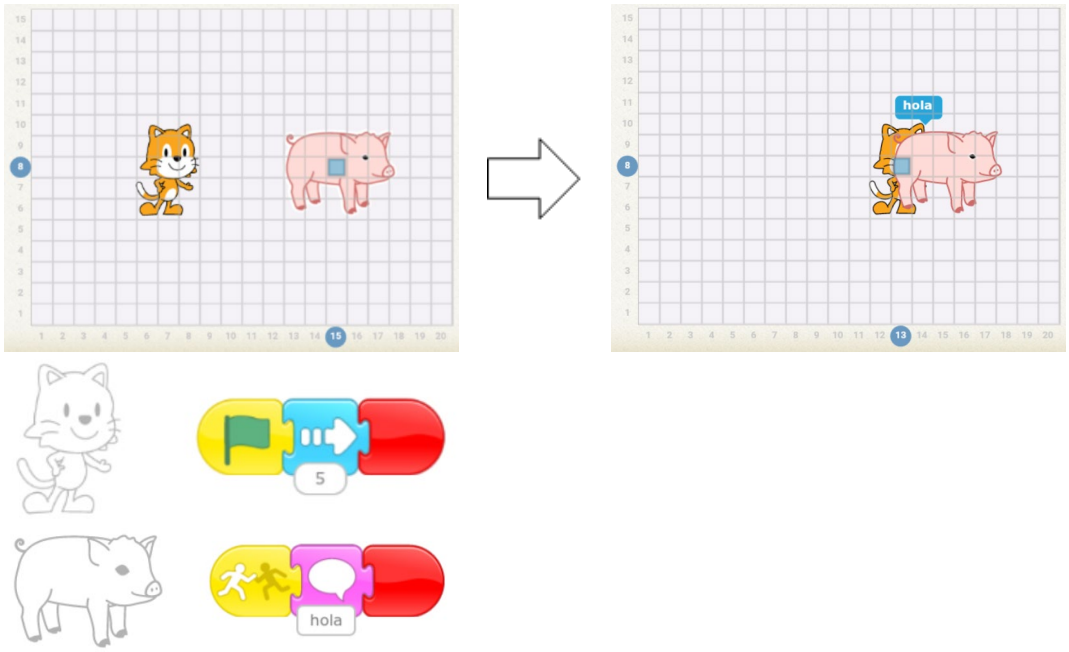
§23.6 Si alguno de los guiones activados por un mensaje no termina su ejecución, tampoco reanuda su ejecución el guion que lo envió (véase el ejemplo 15).

Ejemplos:

Ejemplo 9. El gato se mueve a la derecha, chocando con el cerdo, que se aleja, se vuelve y saluda.



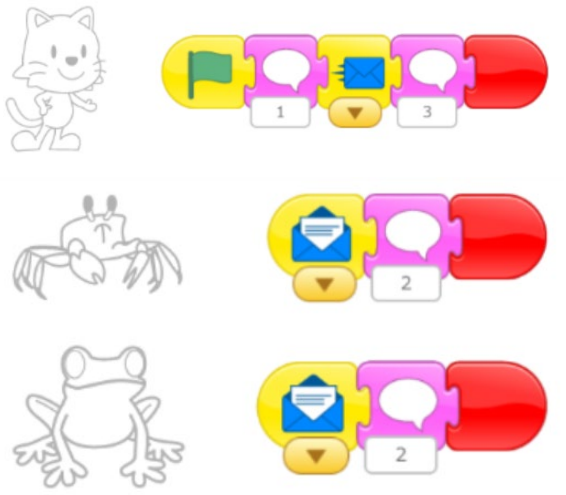
Ejemplo 10. El gato se mueve a la derecha, chocando con el cerdo, que saluda. Al quedarse los dos personajes en contacto, el cerdo se reactiva repetidamente, saludando una y otra vez, hasta que paramos la ejecución del programa (pulsando el icono de hexágono rojo).



Ejemplo 11. Si el programador coloca la serpiente sobre el cacto, el guion de éste se activa, aunque el programador no pretendiera su ejecución. Mientras ambos personajes sigan tocándose, el guion del cacto se reactivará repetidamente.

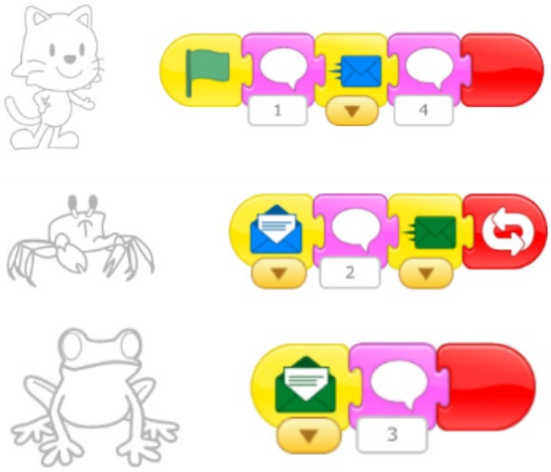
Ejemplo 12. Dos personajes se sincronizan mediante un mensaje, recitando los tres primeros números naturales en orden.

Ejemplo 13. Tres personajes se sincronizan mediante un mensaje, recitando los tres primeros números naturales en orden. Los personajes Cangrejo y Rana dicen el 2 al mismo tiempo.



Ejemplo 14. Tres personajes se sincronizan mediante mensajes, recitando los cuatro primeros números naturales en orden.

Ejemplo 15. Tres personajes se sincronizan mediante mensajes para recitar los primeros números naturales en orden. Sin embargo, los personajes Cangrejo y Rana dicen repetidamente el 2 y 3, mientras que Gato nunca llega a decir el 4.



Conjunto de reglas #5 – Control y terminadores

§24 Los bloques de control permiten controlar el flujo o el tiempo de ejecución de un guion.

§24.1 Hay tres bloques de control, aparte del bloque “Fijar velocidad”, visto en el conjunto de reglas 3:

a. Bloque “Esperar” 

c. Bloque “Parar” 

d. Bloque “Repetir” 

§24.2 Los bloques “Esperar” y “Repetir” tienen un parámetro numérico que puede tomar valores no negativos, incluido el cero.

§24.3 El bloque “Esperar” produce que el guion quede parado durante cierto tiempo (en décimas de segundo).

§24.4 El bloque “Parar” produce que los demás guiones del personaje paren su ejecución, pero el guion en el que se encuentra este bloque continúa su ejecución (véanse los ejemplos 16 y 17).

§25 El bloque “Repetir” produce que los bloques que engloba internamente se ejecuten tantas veces como indica su parámetro numérico. Es un bucle con contador.

§25.1 Un bloque “Repetir” con su parámetro igual a cero ejecuta una vez los bloques que engloba internamente.

§25.2 Un bloque “Repetir” puede englobar cualquier secuencia de bloques que no sean disparadores ni terminadores. También puede englobar otros bloques “Repetir” (véase el ejemplo 18).

§26 Los bloques terminadores son el último bloque de un guion e indican qué hacer a continuación.

§26.1 Hay tres bloques terminadores:

a. Bloque “Finalizar” 

c. Bloque “Repetir indefinidamente” 

d. Bloque “Ir a la página” 

§26.2 El bloque “Finalizar” indica el fin de la ejecución del guion (véase el ejemplo 19).

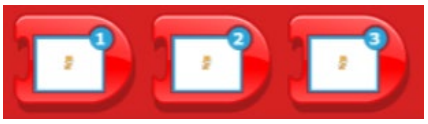
§26.3 El bloque “Repetir indefinidamente” produce que el guion vuelva a ejecutarse desde su comienzo (véase el ejemplo 20). Es un bucle infinito.

§27 El bloque “Ir a la página” produce un cambio de ámbito del programa, pasando a la página indicada.

§27.1 Inicialmente, el bloque “Ir a la página” no aparece en la paleta de bloques, ya que un proyecto ScratchJr tiene inicialmente una sola página.

§27.2 Si se añade una nueva página, aparece automáticamente en la paleta de bloques un bloque “Ir a la página” con el número de la nueva página.

§27.3 Cuando existen varias páginas y se está desarrollando el programa de una de estas páginas, en la paleta de bloques aparecen bloques “Ir a la página” que permiten ir a cualquiera de las otras páginas.



§27.4 Un proyecto ScratchJr admite un máximo de 4 páginas.

§27.5 El bloque “Ir a la página” para la ejecución de todos los guiones de la página actual y cambia a la página indicada en el bloque (véanse los ejemplos 21 y 22).

§27.6 Al ir a una nueva página, cambia el escenario, los personajes y sus programas y sonidos a los de la nueva página, y comienza la ejecución de todos los guiones de la nueva página cuyo bloque disparador sea “Al presionar bandera verde” (véase el ejemplo 23).

§27.7 Al pulsar el icono de bandera verde, comienza la ejecución de los guiones de la página activa que empiezan por el bloque “Al presionar bandera verde”, aunque no sea la primera página del programa.

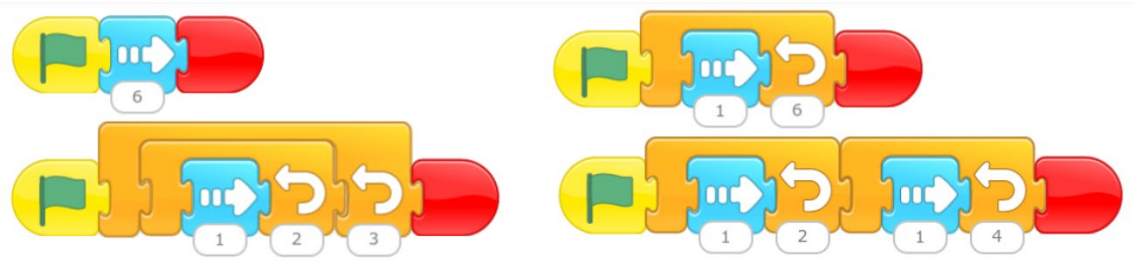
Ejemplos:

Ejemplo 16. Un personaje que está andando continuamente se para cuando el programador le pulsa.

Ejemplo 17. El personaje anda indefinidamente.

El bloque “Parar” no produce ningún efecto sobre el guion en el que se encuentra. Si la intención del programador era que el personaje se parara en algún momento, el bloque “Parar” debe estar en un guion distinto, como en el ejemplo 16.

Ejemplo 18. El personaje da 6 pasos con cualquiera de los siguientes guiones.



Ejemplo 19. Los dos guiones siguientes producen el mismo efecto. Poner un bloque “Finalizar” al final de un guion es opcional, pero es un elemento de buen estilo de programación porque avisa de que el programador da por terminado dicho guion.

Ejemplo 20. Ambos guiones producen un movimiento continuo del personaje. El bloque “Repetir” es innecesario ante la presencia del bloque “Repetir indefinidamente”.

Ejemplo 21. El personaje avanza 5 pasos a la derecha y va a la página 2. Estaba previsto que saltara y girara cuatro veces, pero cambia de página antes de que le dé tiempo a hacerlo.

Ejemplo 22. El personaje realiza varias acciones en paralelo, de movimiento y de hablar. Sin embargo, hay indeterminación sobre qué bloque “Ir a página” se ejecutará primero y, por tanto, si algunas acciones se quedarán sin realizar. Por tanto, en general es un elemento de buen estilo de programación que haya un solo bloque dentro de una página para ir a otra página.

Ejemplo 23. El gato se encuentra saltando repetidamente en el campo. Cuando pasan 5 segundos, va a la página 2, empezando a ejecutar otro guion en el que canta repetidamente.