INTRODUCCIÓN A LA PROGRAMACIÓN CON SCRATCH

1.- ¿QUÉ ES SCRATCH?

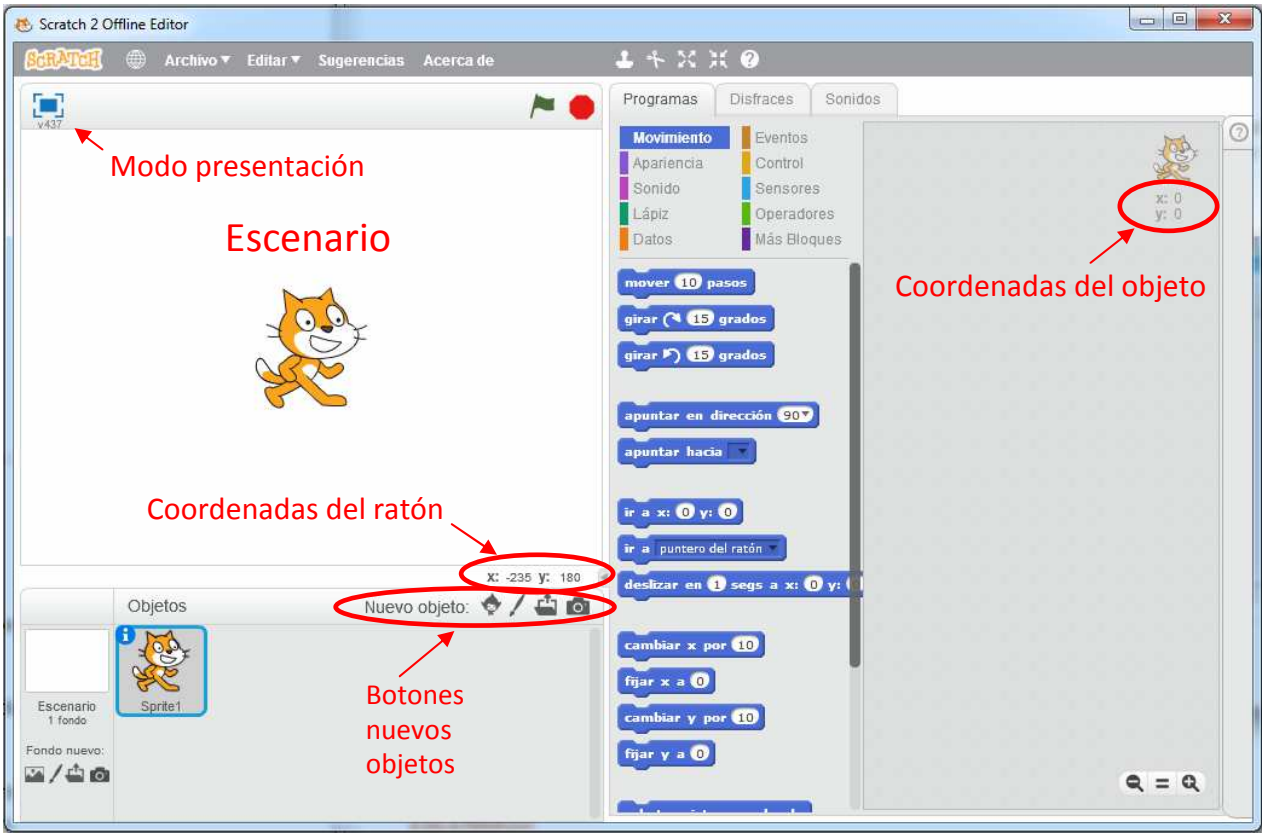
Scratch es un entorno de programación visual y multimedia destinado a la realización de secuencias animadas con o sin sonido y al aprendizaje de la programación. Es software completamente libre. Nosotros trabajaremos con la versión 2.0. Puede trabajarse con el programa descargado en el ordenador o bien online.

La interface es muy intuitiva y permite manipular imágenes, fotos, sonidos, etc, y sobre todo una forma de programación visual con dichos elementos. Con Scratch se comprenden fácilmente conceptos matemáticos e informáticos como los bucles, decisiones condicionales, coordenadas, variables, etc.

Aunque no todos los alumnos van a ser programadores informáticos profesionales, aprender a programar permite experimentar de forma creativa y ayuda al desarrollo del pensamiento lógico y analítico.

2.- LA INTERFACE DE SCRATCH

Cuando ejecutamos el programa nos aparece la interface.



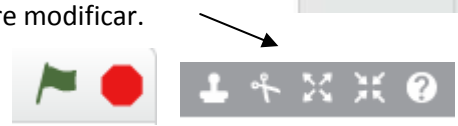
Nos encontramos las siguientes zonas:

- **Escenario**: es la zona de fondo blanco donde está el gato. Aquí es donde se desarrolla la animación que diseñemos. El escenario tiene 480 píxeles de ancho y 360 de alto. El centro del escenario es el punto de coordenadas x = 0, y = 0. Si muevo el ratón por el escenario puedo leer sus coordenadas en la esquina inferior derecha. Podemos mover los objetos haciendo clic sobre ellos y arrastrando.
- Icono **Modo presentación** haciendo clic en él puedo ver el escenario a pantalla completa. 
- **Botones nuevos objetos** (sprites): situados bajo el escenario, nos permiten buscar (en la biblioteca de Scratch o en un archivo guardado en nuestro ordenador) o crear (pintando o echando una foto) nuevos personajes, elementos, etc.

- **Lista de Objetos** (sprites): en esta zona aparecen las miniaturas de todos los sprites que hayamos seleccionado para nuestra animación. Al principio sólo está el gato, pero podemos añadir más con los botones anteriores.

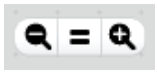
Haciendo clic con el botón derecho del ratón, aparece un menú que permite borrar, duplicar, esconder, o mostrar información del objeto (coordenadas, dirección, etc.).

- **Fondo**: permite cambiar la apariencia del fondo del escenario. Puedo cargar nuevos fondos (desde biblioteca o archivo) o crearlos (pintando o por foto).
- **Barra de herramientas**: permite aumentar o disminuir el tamaño de los objetos, duplicarlos o borrarlos. Se utilizan haciendo clic sobre la herramienta (cambia la forma del puntero) y después clic sobre el objeto que se quiere modificar.



- **Bandera verde y botón rojo**: sirven para ejecutar y detener respectivamente los programas que hayamos creado.
- **Paleta de bloques**: es una caja donde aparecen todos los tipos de instrucciones que podemos utilizar en nuestros programas (scripts). Estas instrucciones se representan en forma de bloques que vamos arrastrando hacia el área de programas, en la que los encajamos en el lugar correcto y ajustamos sus parámetros. Los bloques están clasificados por tipos, disponiendo de diez cajas de bloques diferentes, a cada una de las cuales se le ha asignado un color.

- **Área de programas**: Es la zona situada a la derecha donde vamos montando nuestro programa a base de arrastrar bloques de instrucciones desde la paleta de bloques. Disponemos de unos botones en la parte inferior que nos permite manipular el zoom. Podemos agregar **comentarios** haciendo clic con el botón derecho del ratón en el área de programas. Podemos borrar, duplicar o comentar bloques con clic en el botón secundario sobre el bloque.

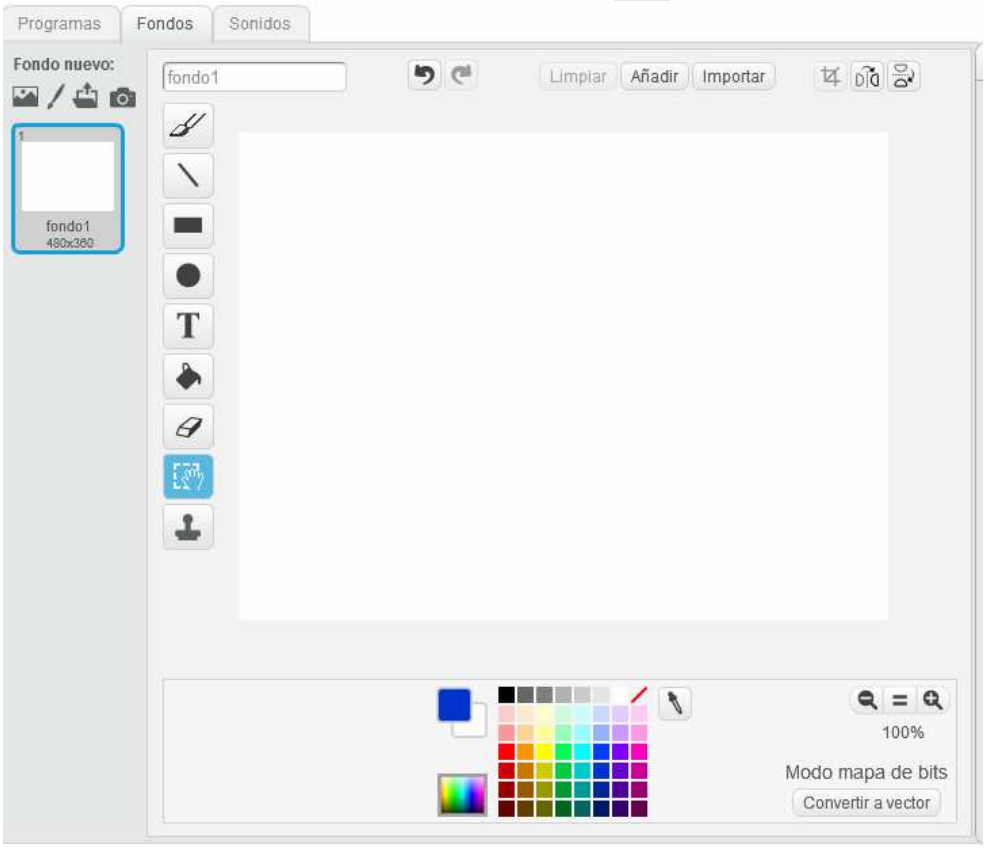


- **Pestañas**: son tres, Programas, Disfraces y Sonidos y nos permiten editar estos tres tipos de elementos. Cada objeto puede tener asociados varios disfraces y sonidos. Se pueden cargar o crear nuevos disfraces y sonidos a los objetos.

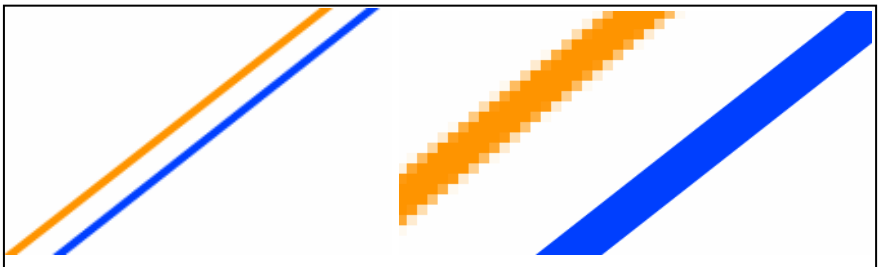
3.- EL EDITOR DE DISFRACES Y FONDOS

Scratch nos permite dibujar nosotros mismos nuevos disfraces para los objetos y fondos para el escenario. También permite editar los que ya tengamos o que hayamos importado desde la biblioteca de scratch o desde nuestro ordenador y retocarlos.

Para ello, entramos en la pestaña Fondos, si vamos a trabajar sobre un nuevo fondo del escenario, o la pestaña Disfraces, si vamos a trabajar con un objeto. Seleccionamos el fondo o disfraz que vayamos a retocar, si ya existe, o bien importamos uno de la biblioteca con  si se trata de un fondo o con  si es un objeto. Si queremos dibujar uno nosotros usamos 



Existen dos técnicas para crear, almacenar y manipular las imágenes digitales: los **mapas de bits** (imágenes JPEG, GIF, PNG) y los **gráficos vectoriales**. Los mapas de bits son conjuntos de puntos con un color y brillo determinados (llamados *píxeles*). Al hacer zoom sobre este tipo de imágenes los píxeles se agrandan con lo que las imágenes se distorsionan (se pixelizan) y pierden calidad. Cuando una imagen es de tipo vectorial, sus elementos individuales (líneas y figuras geométricas) vienen definidos por una ecuación matemática. Cuando ampliamos la imagen vectorial, lo que hace el programa es modificar los parámetros de esta ecuación, lo que hace que la imagen no se *pixelice* y mantiene su calidad. Sin embargo, los gráficos vectoriales son inadecuados para crear imágenes con formas irregulares como paisajes, por ejemplo.



La línea naranja está realizada como mapa de bits y la azul como elemento vectorial. Sin aumento no se aprecian diferencias. Al aumentarlas la naranja se pixeliza y pierde calidad pero la azul no.

Con Scratch podemos dibujar en "modo mapa de bits" o en "modo vector". Podemos pasar de un modo a otro haciendo clic en la esquina inferior derecha sobre "Convertir a vector" o "Convertir a mapa de bits".

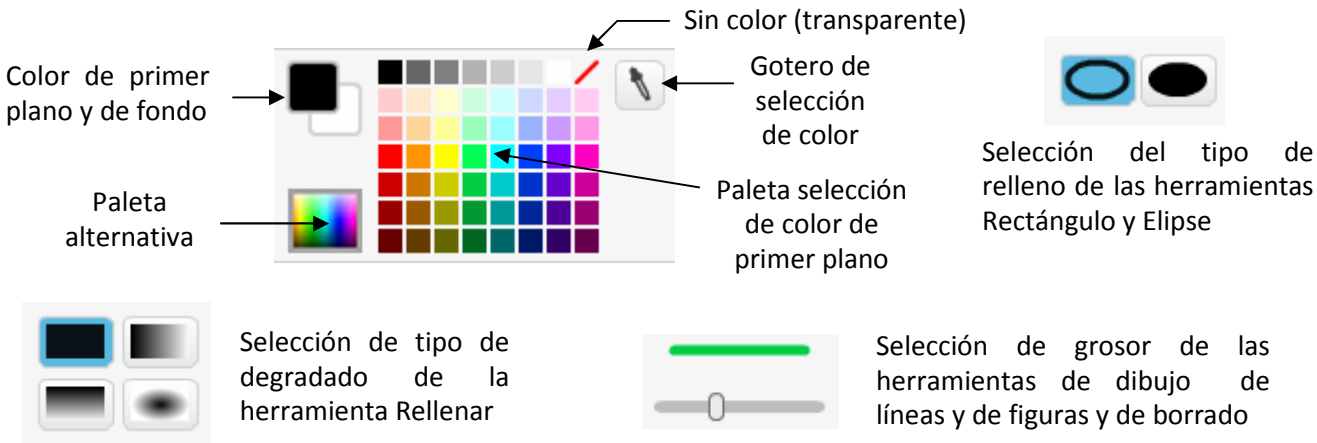
No obstante, ha de tenerse en cuenta que al pasar una imagen de "modo vector" a "modo mapa de bits", las formas vectoriales creadas pierden su carácter vectorial de forma irreversible, es decir, no vuelven a ser formas vectoriales aunque volvamos a pasar al "modo vector".

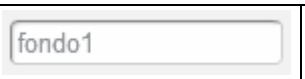
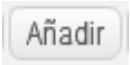
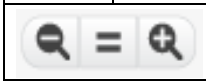
Herramientas específicas del Modo Mapa de bits

En este modo, podemos utilizar para dibujar las siguientes herramientas:

	Brocha: permite pintar libremente usando el color de primer plano manteniendo presionado el botón izquierdo del ratón. Se puede elegir color y grosor en la parte inferior de la pantalla.
	Línea: dibuja una línea recta desde el punto donde se hace clic con el ratón hasta el punto en el que dejo de presionar el ratón. Se puede elegir color y grosor en la parte inferior de la pantalla. Si se mantiene pulsada la tecla "Mayúsculas" la línea se realiza perfectamente horizontal o vertical.
	Rectángulo: Dibuja un rectángulo cuyos vértices opuestos son los puntos en que se hace clic y en el que se deja de presionar el ratón. Puede seleccionarse hueco o relleno y color y grosor de línea abajo. Si se mantiene presionada la tecla "Mayúsculas" se obtiene un cuadrado.
	Elipse: dibuja una elipse dentro del rectángulo cuyos vértices opuestos son los puntos en que se hace clic y en el que se deja de presionar el ratón. Puede seleccionarse hueco o relleno y color y grosor de línea abajo. Si se mantiene presionada la tecla "Mayúsculas" se obtiene un círculo.
	Texto: escribe un texto. Una vez escrito, al hacer clic fuera de la caja que lo enmarca aparecen asas que permiten cambiar el tamaño y/o girarlo. Cuando se vuelve a hacer clic fuera de la caja ya no se pueden realizar cambios. Se puede elegir la fuente del texto y el color en la parte inferior.
	Rellenar con color: rellena la región encerrada por una línea cerrada o el exterior a ella. También regiones limitadas por dos o más líneas cerradas. También cambia el color a formas (líneas, figuras o textos) ya dibujadas. Puede elegirse entre un relleno sólido o bien degradado de diversos tipos. El degradado se hace entre el color de primer plano y el de fondo.
	Borrar: borra con movimientos libres de la mano mientras se mantiene pulsado el ratón. Las áreas borradas quedan transparentes. Se puede seleccionar el tamaño del borrado abajo.
	Seleccionar: selecciona una región rectangular entre el punto donde se hace clic y donde se deja de presionar el ratón. Una vez hecha la selección se puede mover a otro sitio haciendo clic sobre ella y arrastrándola. También se puede cambiar su tamaño o girarla manipulando las asas de la selección. Si una vez hecha la selección se pulsa la tecla "Suprimir" se borra el interior de la selección, mientras que si se pulsa "Mayúsculas + Suprimir" se borra el exterior de la selección.
	Seleccionar y duplicar: selecciona una región rectangular, como el anterior, pero además la duplica, de modo que al arrastrarla lo que se arrastra en sólo la copia. Al igual que el comando anterior también permite girar o cambiar el tamaño de la selección.
	Recortar: Recorta el contenido del lienzo a la selección actual, dejando el resto del lienzo transparente.

Herramientas comunes a ambos modos



	**Nombre del disfraz o fondo**: En este cuadro de texto podemos modificar el nombre del disfraz o fondo de escenario
	**Deshacer y rehacer**: Deshace y rehace respectivamente, la última operación realizada. Pueden aplicarse sucesivas veces para revertir varias operaciones.
	**Limpiar**: Elimina todo el contenido del lienzo. Lo deja transparente si se trataba de un disfraz o blanco si se trataba de un fondo del escenario.
	**Añadir**: Añade un elemento desde la biblioteca de Scrath, bien un disfraz de objeto o un fondo de escenario.
	**Importar**: Añade una imagen desde nuestro ordenador.
	**Voltear**: Voltea horizontal y verticalmente, respectivamente, la selección que tengamos marcada. Si no hay ninguna selección marcada voltea todo el lienzo.
	**Fijar centro**: Fija el centro del disfraz. No está operativo cuando editamos un fondo.
	**Zoom**: disminuye, pone a tamaño real y aumenta respectivamente el tamaño del lienzo (como para verlo desde más cerca o más lejos) pero no cambia el tamaño de la imagen.

Herramientas específicas del Modo Vector

Cuando trabajamos en el "modo vector" cada forma (línea, figura o texto) que creamos se dibuja en una **capa**. Una capa podemos suponerla como una hoja de papel transparente donde se dibuja la forma. La primera forma creada ocupa la capa situada más abajo y la última creada se dibuja en la capa situada más arriba. El orden de las capas determina qué forma tapa a otra. Cada capa tapa a las situadas debajo de ella y es tapada por las que se sitúan por encima. Como veremos, podremos cambiar el orden de las capas.

Las herramientas específicas del modo vector (aparecen en el lado derecho del lienzo) son:

	**Seleccionar**: selecciona formas (líneas, figuras y textos) al hacer clic sobre ellos. Si vamos haciendo clic sobre varias formas con la tecla "Mayúsculas" o la tecla "Control" pulsadas se van seleccionando todas ellas y si volvemos a hacer clic sobre una forma ya seleccionada, se deselecciona. Si hacemos clic y arrastramos formando una ventana de selección que englobe a varias formas se seleccionan todas las que estén totalmente incluidas dentro de la ventana. Una vez creada la ventana de selección podemos cambiar su tamaño o girarla tirando de las asas, lo que afectará a todas las formas seleccionadas. Podemos desplazar la selección haciendo clic en un punto perteneciente o interior a alguna de las formas seleccionadas y arrastrando. Se cancela la selección haciendo clic en un punto fuera de cualquiera de las formas.
	**Reformar**: al hacer clic sobre una forma aparecen *puntos de ajuste* (por ejemplo, vértices) los cuales podemos seleccionar haciendo clic sobre ellos y arrastrarlos para cambiar la forma. También podemos crear nuevos puntos de ajuste haciendo clic en los lugares deseados de las líneas. También podemos eliminar puntos de ajuste existentes haciendo clic sobre ellos y dejando de hacer clic sin haberlos arrastrado.
	**Lápiz**: permite pintar libremente usando el color de primer plano manteniendo presionado el botón izquierdo del ratón. Se puede elegir color y grosor en la parte inferior de la pantalla..
	**Línea**: dibuja una línea recta desde el punto donde se hace clic con el ratón hasta el punto en el que dejo de presionar el ratón. Se puede elegir color y grosor en la parte inferior de la pantalla. Si se mantiene pulsada la tecla "Mayúsculas" la línea se realiza perfectamente horizontal o vertical.
	**Rectángulo**: Dibuja un rectángulo cuyos vértices opuestos son los puntos en que se hace clic y en el que se deja de presionar el ratón. Puede seleccionarse hueco o relleno y color y grosor de línea abajo. Si se mantiene presionada la tecla "Mayúsculas" se obtiene un cuadrado.

	**Elipse**: dibuja una elipse dentro del rectángulo cuyos vértices opuestos son los puntos en que se hace clic y en el que se deja de presionar el ratón. Puede seleccionarse hueco o relleno y color y grosor de línea abajo. Si se mantiene presionada la tecla "Mayúsculas" se obtiene un círculo.
	**Texto**: escribe un texto y permite editar un texto previamente escrito. Para cambiar el tamaño o girar un texto es preciso seleccionarlo con la herramienta de selección. es Se puede elegir la fuente del texto y el color en la parte inferior.
	**Colorear formas**: colorea formas (líneas, figuras y textos), tanto su interior (si son formas cerradas) como su borde haciendo clic en su interior o sobre su borde respectivamente. Puede elegirse entre un relleno sólido o varios tipos de degradado.
	**Duplicar**: Duplica una forma al hacer clic sobre ella. A continuación, sin dejar de presionar el ratón, podemos arrastrar la copia a otro lugar y luego podemos cambiar su tamaño o girarla tirando de las asas. Si tras hacer clic sobre la forma que queremos duplicar con la herramienta "duplicar" seleccionada, mantenemos la tecla "Mayúsculas" presionada, podemos ir haciendo una copia de la forma en cada uno de los lugares en los que vayamos haciendo clic (copias múltiples)
	**Una capa hacia delante**: aparece la herramienta sólo si hay seleccionada alguna forma. Al hacer clic sobre esta herramienta sitúa la forma seleccionada una capa hacia arriba. Si mantenemos pulsada la tecla "Mayúsculas" al hacer clic sobre la forma seleccionada la trae al frente, es decir, a la capa situada más arriba de todas.
	**Una capa hacia atrás**: aparece la herramienta sólo si hay seleccionada alguna forma. Al hacer clic sobre esta herramienta, sitúa la forma seleccionada una capa hacia abajo. Si mantenemos pulsada la tecla "Mayúsculas" al hacer clic sobre la forma seleccionada la envía al fondo, es decir, a la capa situada más abajo de todas.
	**Grupo**: aparece la herramienta sólo cuando hay más de una forma seleccionada. Si hacemos clic sobre la herramienta crea un grupo con las formas seleccionadas. Cuando varias formas forman un grupo se desplazan a la vez, cambia de capa a la vez, cambian de tamaño o giran a la vez, etc. Un grupo puede formar parte de otro grupo mayor (que engloba a más formas) y así sucesivamente.
	**Desagrupar**: aparece la herramienta sólo cuando hay algún grupo seleccionado. Al hacer clic sobre la herramienta el grupo se deshace y las formas que lo componían vuelven a manipularse por separado. Cuando un grupo está integrado a su vez por otros grupos más pequeños, al aplicar la herramienta desagrupar, el grupo grande se disgrega pero los grupos más pequeños que lo integraban siguen formados. Hay que aplicar la herramienta sobre estos grupos para disgregarlos.

Trabajar en el Modo Vector tiene la enorme ventaja de que cada forma es independiente del resto del dibujo, con lo que en todo momento podemos modificarlas, trasladarlas de un lugar a otro, borrarlas, editar los textos (cambiar el texto o el tipo de letra), etc., con mucha facilidad, cosa que resulta difícil en modo mapa de bits.

Por ello, es importante que recordemos, como hemos dicho antes, que el cambio del Modo Vector al Modo Mapa de bits hace que las formas creadas en el modo vector pierdan su carácter vectorial de forma irreversible, y ya no se recupera dicho carácter aunque pasemos de nuevo al modo vector.

Resulta conveniente, por tanto, planificar bien el orden de realización de un dibujo. Conviene empezar en Modo Mapa de bits para realizar las partes del dibujo que no podamos hacer en Modo Vector (por ejemplo, la incorporación de imágenes a las que tengamos que borrarles alguna parte o retocarlas), y luego pasar a Modo Vector para incorporar las formas más regulares.

4.- EVENTOS DE INICIO DE PROGRAMAS

Existen diversos eventos que pueden dar lugar al inicio de un programa: al hacer clic sobre la bandera verde, al presionar una tecla, al hacer clic sobre un objeto, al recibir un mensaje, etc. Todos estos elementos están en el grupo de bloques Eventos.

Nosotros empezaremos utilizando la bandera verde.

Por ejemplo, con el programa de la derecha al hacer clic sobre la bandera verde se situará al gatito en el punto de coordenadas absolutas (0,0) y, tras esperar 1 segundo, lo desplazará a las coordenadas absolutas (180, 100).

5.- MOVIMIENTO DE LOS OBJETOS

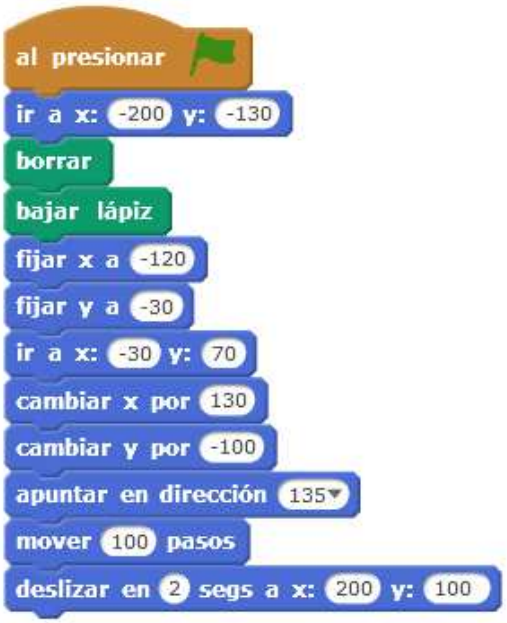
Si construyes y ejecutas el programa anterior observarás que el movimiento del gato desde unas coordenadas a otras es casi instantáneo y no podemos visualizarlo.

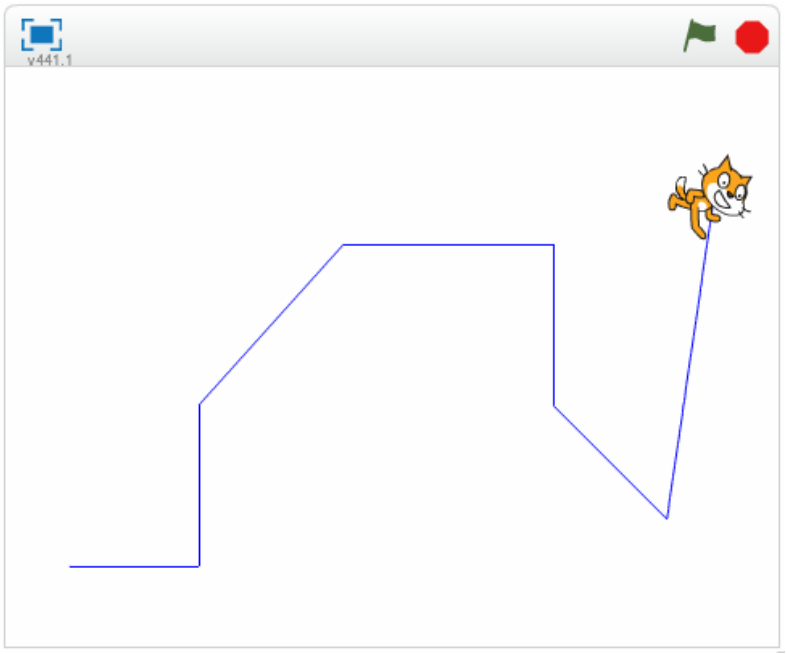
Para poder ver el recorrido que hace el gato, vamos a hacer que pinte en la pantalla al moverse. También haremos que borre la pantalla cada vez que inicie el programa para eliminar los rastros de la ejecución anterior del programa. Usaremos los bloques “bajar lápiz” y “borrar”. Observa el recorrido del gato.

Disponemos de diversos bloques para hacer movimientos (más adelante veremos más):

	Desplaza el objeto a las coordenadas **X** e **Y** indicadas
	Desplaza el objeto de modo que la coordenada **X** adopta el valor indicado en el bloque mientras que la coordenada **Y** se queda como está en ese momento.
	Desplaza el objeto de modo que la coordenada **Y** adopta el valor indicado en el bloque mientras que la coordenada **X** se queda como está en ese momento.
	Desplaza el objeto de modo que la coordenada **X** se incrementa en el valor indicado en el bloque mientras que la coordenada **Y** se queda como está.
	Desplaza el objeto de modo que la coordenada **Y** se incrementa en el valor indicado en el bloque mientras que la coordenada **X** se queda como está.
	Apunta el objeto en la dirección especificada (0º hacia arriba, 90º hacia la derecha, 180º hacia abajo, -90º hacia la izquierda.
 	Rota el objeto en el sentido de la flecha el ángulo especificado.
	Mueve el objeto hacia delante (valores positivos) o hacia atrás (valores negativos) los pasos especificados en la dirección actual del objeto.
	Desplaza el objeto hasta las coordenadas indicadas pero invirtiendo el tiempo especificado.

Ejemplo 1:





Observa cómo todos los tramos, menos el último, se realizan de forma casi instantánea. El último se hace de una forma más lenta, que podemos visualizar. Esto es gracias a que hemos indicado que se invierta un tiempo en realizar el recorrido. Cuando mayor el tiempo más lento se mueve el objeto.

6.- EL LÁPIZ

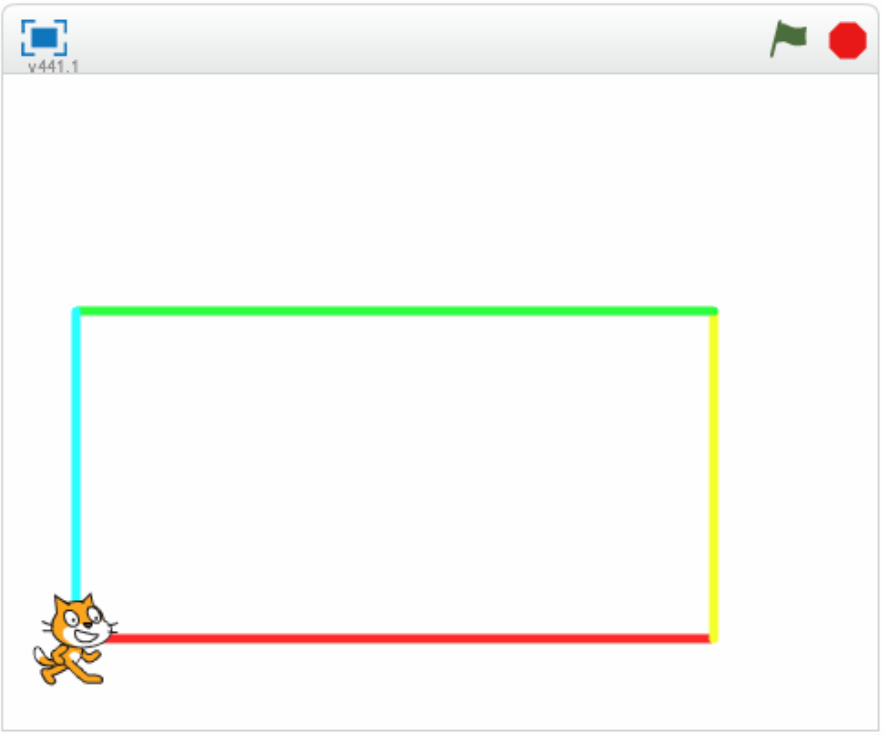
Podemos imaginar que los objetos portan un lápiz muy grande con la punta mirando hacia abajo. Cuando el objeto baja el lápiz, la punta de éste va rozando por el suelo y deja pintado el recorrido que el objeto realiza. Cuando el objeto levanta el lápiz, no roza con el suelo y no pinta.

Existen varios bloques para modificar las características del lápiz: tamaño (grosor), color e intensidad.

bajar lápiz	Dibuja al moverse	subir lápiz	No dibuja al moverse	borrar	Borra todas las marcas de lápiz que haya en escenario

fijar color de lápiz a	El lápiz cambia al color indicado. Se cambia el color haciendo clic sobre el cuadradito y a continuación sobre algo del color deseado.
fijar color de lápiz a 100	Establece el color del lápiz al valor numérico indicado. El rango va desde 0 a 200.(0 rojo, 34 amarillo, 70 verde, 130 azul, 170 magenta)
fijar tamaño de lápiz a 10	Establece el tamaño (grosor) del lápiz al valor indicado.
fijar intensidad de lápiz a 50	Establece la intensidad del lápiz al valor indicado. Rango 0 - 100. Valores pequeños muy oscuro, valores altos muy claro.
cambiar color del lápiz por 10	Aumenta (valores positivos) o disminuye (valores negativos) el color del lápiz el valor indicado.
cambiar tamaño de lápiz por 1	Aumenta (valores positivos) o disminuye (valores negativos) el grosor del lápiz el valor indicado.
cambiar intensidad de lápiz por 10	Aumenta (valores positivos) o disminuye (valores negativos) la intensidad del lápiz el valor indicado.

Ejemplo 2:



7.- LOS BUCLES REPETITIVOS

En algunas ocasiones necesitamos que un mismo grupo de instrucciones se ejecuten varias veces, bien un número determinado de veces, o de forma permanente o bien hasta que se cumpla una condición.

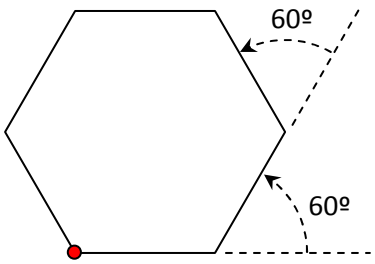
	Ejecuta el número de veces especificado los bloques de instrucciones contenidos en su interior.		Ejecuta continuamente los bloques de instrucciones contenidos en su interior.
	Comprueba si la condición es falsa; si lo es, ejecuta los bloques de instrucciones contenidos en su interior y vuelve a comprobar la condición y a ejecutar las instrucciones, y así sucesivamente. Cuando la condición es verdadera pasa a los bloques siguientes.		

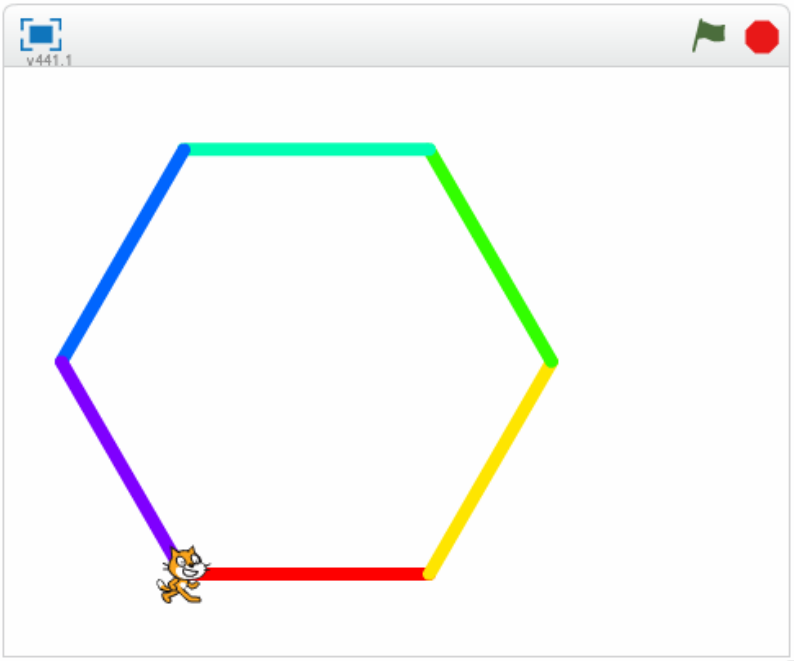
Ejemplo 3: el gato estará dando saltitos hasta que se pulse la tecla "p" y entonces parará (hay que mantener pulsada la tecla "p" un tiempo suficiente para garantizar la parada).

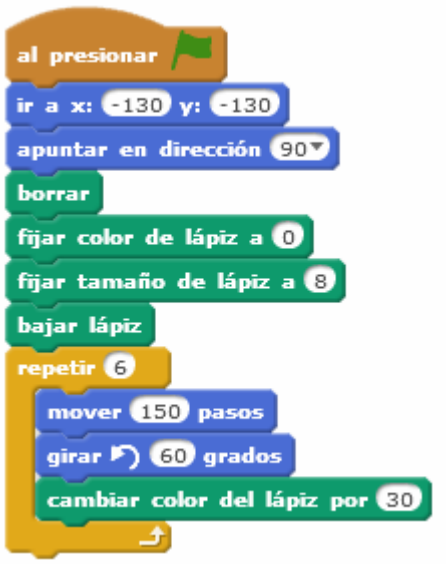
Ejemplo 4: el gato dará 5 saltitos y entonces parará.

Ejemplo 5: el gato estará dando saltitos de forma permanente hasta que paremos el programa haciendo clic en el botón rojo.

Ejemplo 6: Construcción de un hexágono valiéndonos de la información geométrica de la figura. El punto de inicio será el vértice marcado con un circulito rojo. Cada lado será de un color diferente. El tamaño del lápiz de 8 puntos.





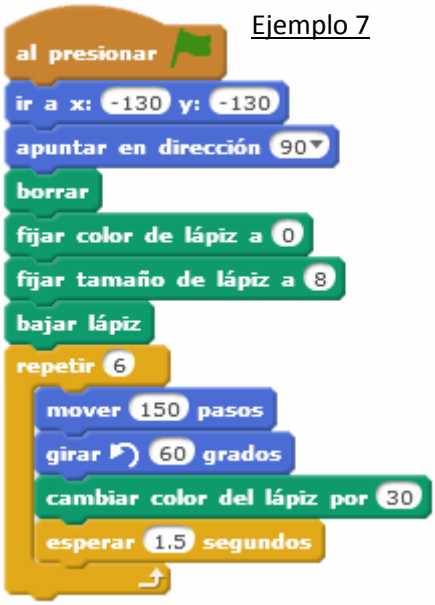


8.- ESPERAS DE TIEMPOS O DE EVENTOS

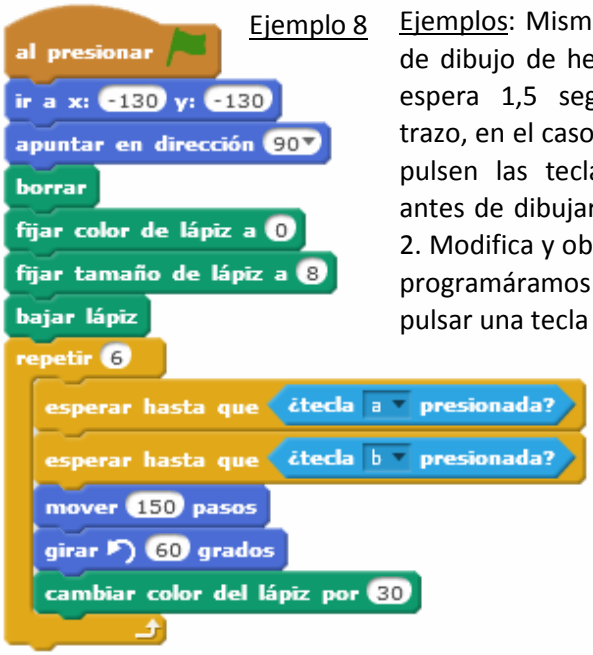
En algunas ocasiones necesitamos que nuestros programas aguarden a la espera durante un tiempo determinado o bien hasta que se cumpla alguna condición.

	Espera el número de segundos especificado y continúa con el bloque siguiente. El número de segundos puede ser un número decimal (por ejemplo, 0.3)
	Espera hasta que la condición indicada sea verdadera y entonces ejecuta los bloques que siguen.

Ejemplo 7



Ejemplo 8



Ejemplos: Mismos ejercicios anteriores de dibujo de hexágono, pero ahora se espera 1,5 segundos entre trazo y trazo, en el caso 1, y se espera a que se pulsen las teclas “a” y después “b” antes de dibujar cada trazo, en el caso 2. Modifica y observa lo que ocurriría si programáramos que sólo hubiera que pulsar una tecla en vez de dos.

9.- LA APARIENCIA DE LOS OBJETOS

Los objetos pueden cambiar su apariencia, lo que incluye mostrarse, esconderse, cambiar de tamaño, de color, brillo, etc.

mostrar	Hace aparecer el objeto en el escenario	esconder	Hace desaparecer el objeto del escenario. Un objeto escondido no puede ser detectado por otros objetos con el bloque ¿tocando ?
establecer efecto color a 0	Establece el efecto gráfico seleccionado (color, brillantez, desvanecer, etc.) en el valor indicado.		
cambiar efecto color por 25	Cambia (aumenta si valor positivo o disminuye si valor negativo) el efecto seleccionado en el valor indicado.		
quitar efectos gráficos	Quita los efectos gráficos anteriores que tenga aplicados (afecta al color, la brillantez, etc., pero no al tamaño		
fijar tamaño a 100 %	Establece el tamaño del objeto al valor indicado.		
cambiar tamaño por 10	Aumenta (valores positivos) o disminuye (valores negativos) el tamaño del objeto en % el valor indicado.		

Los objetos también, pueden cambiar de disfraz. Los disfraces de un objeto son las diferentes imágenes con las que puede mostrarse, pero la diferencia entre disfraces no tiene que limitarse a cambios de tamaño o color, sino que puede tratarse de diferentes posiciones de un mismo personaje, para dar la sensación de movimiento al ir cambiando con rapidez una por otra (como los GIF animados) o incluso tratarse de imágenes diferentes.

Por ejemplo, el gatito tiene dos disfraces. Al cambiar rápidamente uno por otro parece que camina. La bailarina tiene cuatro y al ir cambiando uno por otro rápidamente parece que baila.

cambiar disfraz a disfraz2	Cambia el disfraz actual por el seleccionado.
siguiente disfraz	Cambia el disfraz actual por el siguiente de la lista de disfraces. Cuando llega al último vuelve al primero.

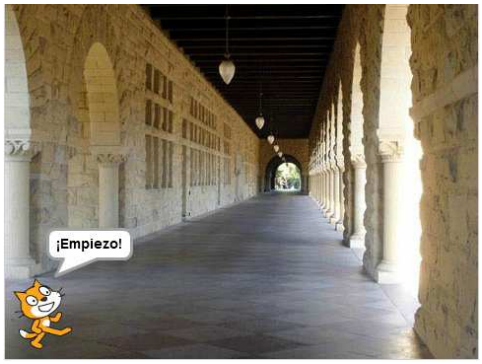
Los objetos también pueden emitir mensajes en forma de burbujas de diálogo o de pensamiento.

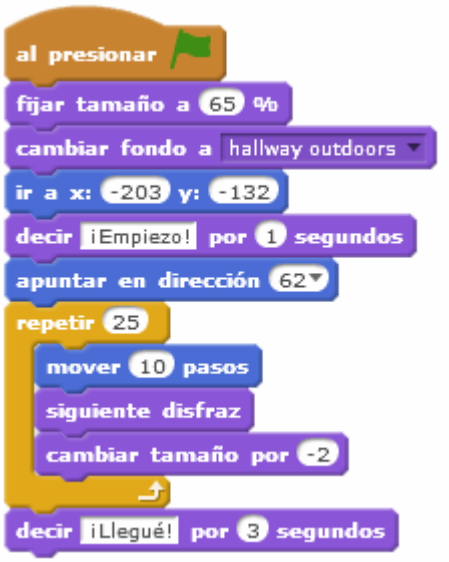
decir Hola! pensar Bien!	Despliega una nube de diálogo o de pensamiento del objeto de forma permanente. Esta nube se elimina si se ejecuta el bloque sin texto.	
decir Hola! por 2 segundos	pensar Bien! por 2 segundos	Despliega la nube de diálogo o de pensamiento por el tiempo indicado.

El **escenario** es, prácticamente, como un objeto más: puede tener programas asociados; puede tener varios disfraces, que en este caso se denominan “fondos”; se le pueden aplicar efectos gráficos como efectos de color, de brillo, etc. Para añadir nuevos fondos al escenario podemos usar varias herramientas, desde cargarlos desde la biblioteca, a cargarlos desde un archivo de imagen o pintarlos nosotros mismos.

Ejemplo 9:

Carga desde la biblioteca el fondo llamado “hallway outdoors” y, a continuación, construye y ejecuta para el gatito el programa de la figura. Observarás que el gato empieza al 65% de su tamaño en la esquina inferior izquierda del escenario, dice el mensaje “¡Empiezo!” y empieza a moverse cambiando de disfraz en dirección 62º al mismo tiempo que se va reduciendo su tamaño para parecer que se está alejando. Al final del trayecto emite el mensaje “¡Llegué!”. En cuanto al escenario, aunque inicialmente estemos con el fondo blanco por defecto, el propio programa del gato hace que cambie el fondo al que le indicamos.





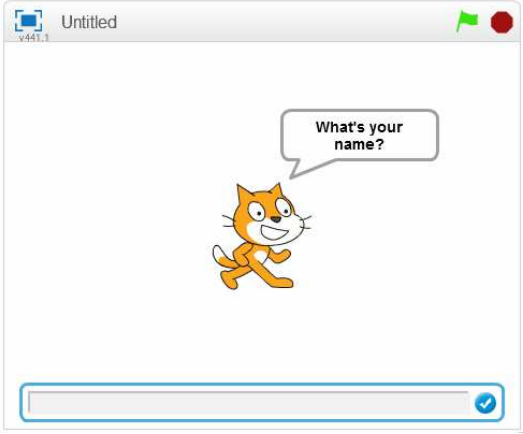
10.- LA ENTRADA DE DATOS DESDE EL TECLADO

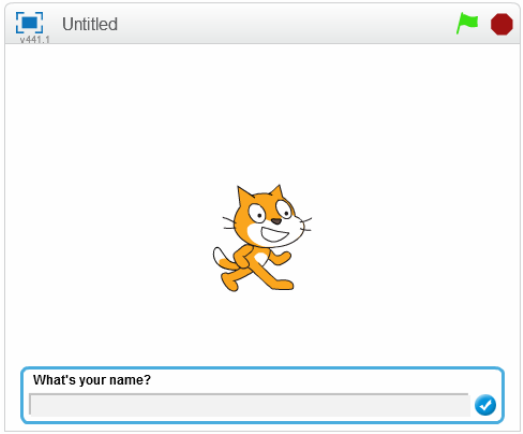
En muchas ocasiones necesitamos suministrarle datos a los programas. Por ejemplo, podemos pensar en un programa que nos dibuja polígonos regulares, y nos tiene que preguntar el número de lados y la longitud del lado del polígono.

Los objetos (incluido el escenario) pueden hacer preguntas al usuario para que éste las introduzca a través del teclado.

preguntar What's your name? y esperar	Formula una pregunta en la pantalla y guarda lo que introduzcamos en respuesta . Hace que el programa espere hasta presionar “Enter” o hacer clic en la casilla de verificación.
respuesta	Almacena la entrada por teclado más reciente tras la ejecución del bloque anterior por parte de cualquier objeto o del escenario. Es decir, es compartido por todos los objetos.

Cuando es un objeto el que formula la pregunta ésta aparece en una nube de diálogo, mientras que aparece en la parte inferior de la pantalla cuando pregunta el escenario.

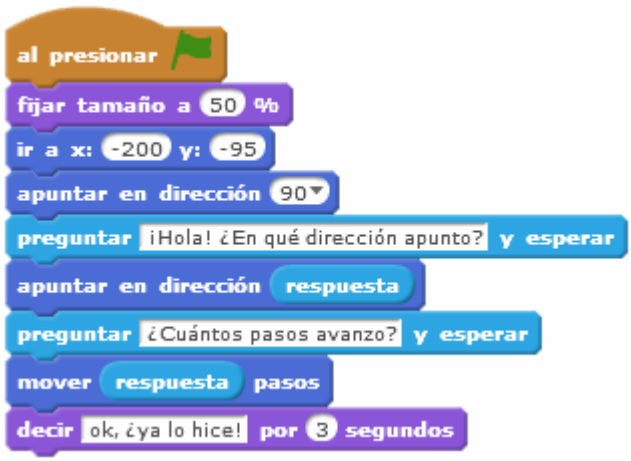




Ejemplo 10:

Construye y ejecuta para el gatito el programa de la figura. Observa como la "respuesta" a las preguntas se puede introducir en los huecos destinados a los datos que hay en los bloques.

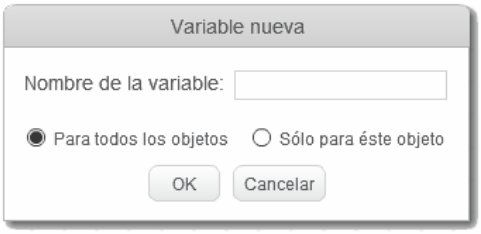
Observa también como tras la primera pregunta, el dato que hay en el bloque "respuesta" es el valor del ángulo de dirección que hayamos introducido, pero después de la segunda pregunta lo que hay en "respuesta" es ya el número de pasos que debe moverse. O sea, solo se guarda el valor más reciente.



11.- EL USO DE LAS VARIABLES

Una variable podemos decir que es un espacio en la memoria con un nombre. En dicho espacio podemos guardar un dato, que puede ser un número o una cadena de caracteres (palabras o frases).

Para crear variables abrimos el grupo de bloques Datos y hacemos clic sobre Crear una variable. En el cuadro de diálogo que se abre indicamos el nombre de la variable y seleccionamos si queremos que sea accesible por todos los objetos o solo para el objeto activo en el momento actual.



Dato	Contiene el dato guardado en la variable y podemos usar este bloque dentro de los huecos redondeados o rectangulares para ingreso de información de otros bloques.
Dato	Clicando sobre la casilla de verificación podemos monitorizar o no en pantalla el valor actual de la variable. Puede activarse o desactivarse durante la ejecución del programa.
fijar Dato a 0	Establece el valor de la variable seleccionada en el menú desplegable al valor indicado.
cambiar Dato por 1	Incrementa (si el valor es positivo) o disminuye (si el valor es negativo) el valor de la variable seleccionada en el número indicado.
mostrar variable Dato / esconder variable Dato	Muestra u oculta el valor de la variable en el escenario desde el programa en ejecución.

El valor de la variable se puede monitorizar en pantalla con diferentes formatos. Se cambia de uno a otro haciendo doble clic o haciendo clic con el botón derecho del ratón y seleccionado el formato.

Ejemplo 11: El gato pregunta cuántos saltitos debe dar y guarda el dato en la variable "Número saltos". A continuación repite tantas veces como indique el número introducido el conjunto de bloques que producen el efecto del salto. Cada vez que da un salto dice el número del salto dado, el cual se guarda en la variable "Saltos dados", la cual se incrementa en 1 con cada salto. Al finalizar dice ¡Terminé!

Ejemplo 12: Programa que dibuja polígonos regulares. Para ello, nos pide el número de lados y la longitud del lado del polígono. Ten en cuenta que si la longitud del lado que introducimos es demasiado grande el polígono puede no entrar en la pantalla y se realiza mal.

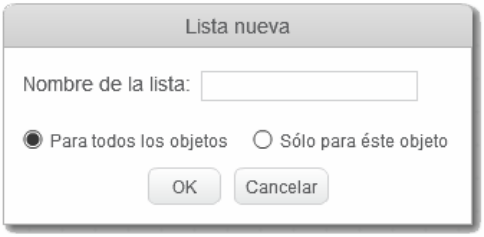
Hemos creado dos variables: "Número lados" y "Longitud lado".

Observa que hemos utilizado un operador matemático, el de la división, que lo veremos en breve.

12.- LAS LISTAS

Una lista es una variable múltiple (contiene varias variables simples). Es como si fuera un cajón con varios compartimentos. Cada compartimento es una variable simple y en él se puede guardar un dato (un número o una cadena de caracteres). Todas las variables simples tienen el mismo nombre (el que le demos a la lista) y se diferencian por un número que indica el lugar que ocupan en la lista (a este número se le llama índice).

Para crear una lista se procede igual que con las variables, abrimos el grupo de bloques Datos y hacemos clic sobre Crear una lista, y damos su nombre y para quién es accesible. Tener cuidado a elegir los nombres de las listas pues, a diferencia de las variables simples, posteriormente no podremos cambiarlos.

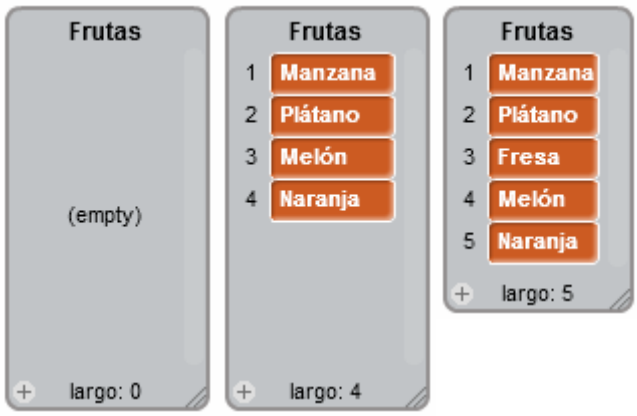


Datos	Contiene todos los datos guardados en la lista y podemos usar este bloque dentro de los huecos redondeados o rectangulares para ingreso de información de otros bloques.
elemento 1 de Datos	Contiene el elemento indicado en el hueco de la lista seleccionada. Puede incluirse en los huecos de otros bloques.
Datos	Clicando sobre la casilla de verificación podemos monitorizar o no en pantalla el contenido actual de la lista. Puede activarse o desactivarse durante la ejecución del programa.
añade thing a Datos	Añade lo que incluyamos en el hueco como un elemento adicional que se coloca en el último lugar de la lista seleccionada.
insertar thing en 1 de Datos	Inserta lo incluido en el hueco como el elemento indicado de la lista. Los elementos con este índice o superior se desplazan hacia delante.
borrar 1 de Datos	Borra el elemento cuyo índice es el número indicado de la lista seleccionada. Los elementos de índice superior al indicado se desplazan hacia atrás en la lista
reemplazar elemento 1 de Datos con thing	Reemplaza el elemento indicado de la lista seleccionada por lo que incluyamos en el hueco.
mostrar lista Datos / esconder lista Datos	Muestra u oculta el contenido de la lista en el escenario desde el programa en ejecución.
longitud de Datos	Contiene el número total de elementos que tiene actualmente la lista seleccionada.
Datos contiene thing ?	Este bloque proporciona un valor lógico "verdadero" si la lista contiene algún elemento que coincida exactamente con lo indicado en el hueco.

Los bloques anteriores permiten modificar (añadir, borrar, insertar, reemplazar), consultar y mostrar o esconder las listas desde los programas. Por ejemplo, programas que van construyendo listas a partir de unos datos que va suministrando el usuario. No obstante, si nuestro programa tiene que partir de unas listas ya construidas desde el momento inicial, podemos hacerlo desde el monitor de lista que podemos mostrar en el escenario.

En el momento de crear la lista, este monitor aparecerá vacío y con longitud 0. Podemos hacer:

- Si le damos al icono + situado en la esquina inferior izquierda podemos añadir nuevos elementos.
- Haciendo clic sobre un elemento y clic en la X lo eliminamos.
- Si seleccionamos un elemento haciendo clic sobre él y a continuación en + insertamos un nuevo elemento tras el que hemos seleccionado.
- Para cambiar el valor de un elemento solo tenemos que seleccionarlo haciendo clic y cambiar el contenido.
- Podemos redimensionar el tamaño del monitor de lista desde la esquina inferior derecha.



Ejemplo 13: Vamos a partir de una lista ya hecha con las provincias de Andalucía, pero con varios errores (provincias que no son, otras que faltan, alguna repetida). Haremos un programa que corrija los errores y deje bien la lista y en orden alfabético, practicando algunos de los bloques que hemos visto. Como mostraremos la lista en el escenario podemos ir viendo los cambios conforme se van haciendo, por lo que pondremos un tiempo de espera entre un cambio y otro para que nos dé tiempo a visualizarlos. Debemos tener cuidado pues cada vez que hacemos operaciones de borrado o de inserción, se producen cambios en los índices de los elementos.

Observamos que los errores son: sobran Cuenca y Segovia, faltan Cádiz, Huelva y Sevilla, Granada está repetida. Las operaciones que haremos serán:

- Insertar Cádiz entre Almería y Córdoba (esto hace que los índices de Córdoba y las que están debajo aumenten en 1).
- Borrar Cuenca (esto hace que los índices de las que están debajo disminuyan en 1).
- Reemplazar Segovia por Huelva.
- Borrar Sevilla de donde está
- Borrar la segunda aparición de Granada.
- Añadir Sevilla al final.

13.- LOS OPERADORES MATEMÁTICOS, RELACIONALES Y LÓGICOS

OPERADORES MATEMÁTICOS

Scratch dispone de una serie de **operadores matemáticos** para realizar todo tipo de operaciones matemáticas con números o variables que contengan números. Las cadenas de caracteres se evalúan como 0 en los bloques de operaciones matemáticas.

	Suma, resta, multiplicación y división convencionales.
	Proporciona el resto (a lo que se llama módulo) de la división del primer número entre el segundo.
	Proporciona el número entero más cercano al número dado. En caso de equidistancia redondea al más alto. Por ejemplo, 2.5 sería redondeado a 3.
	Proporciona un número entero aleatorio entre los dos valores indicados ambos inclusive. También puede dar el 0.
	Proporciona el resultado de aplicar la operación seleccionada del menú sobre el valor numérico indicado.

Las operaciones matemáticas científicas que se realizan con el último bloque de la tabla anterior son:

Función	Operación /Ejemplo
abs	Valor absoluto. Ej.: abs(9) = 9, abs(– 7)=7
piso	Número entero más próximo pero inferior al valor dado. Ej.: piso(6.8)=6, piso(8.3)=8
techo	Número entero más próximo pero superior al valor dado. Ej.: techo(6.8)=7, techo(8.3)=9
raíz cuadrada	Raíz cuadrada de un número. Ej.: raíz cuadrada(9)=3. Solo da el valor positivo
sin	Función trigonométrica seno. Ej.: sin(60)=0.87.
cos	Función trigonométrica coseno. Ej.: cos(60)=0.50.
tan	Función trigonométrica tangente. Ej.: tan(60)=1.73.
asin	Función trigonométrica arcoseno. Ej.: asin(0.92)=66.93.
acos	Función trigonométrica arcococoseno. Ej.: acos(0.7)=45.57.
atan	Función trigonométrica arcotangente. Ej.: atan(1)=45.
En	Logaritmo neperiano. Proporciona el número al que hay que elevar el número “e” para obtener el número al que se aplica. Ej.: En(25)=3.22.
log	Logaritmo en base 10 de un número. Ej.: log(1000)=3.
e^	Exponencial de un número. Ej.: e^(2)=7.39.
10^	Potencia de 10 de un número. Ej.:10^(4)=10000.

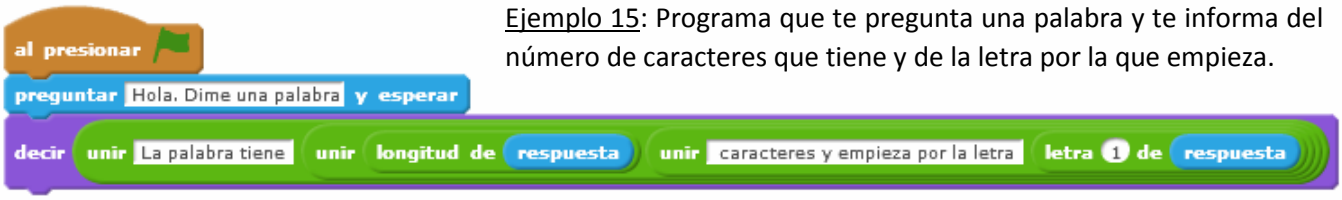
Ejemplo 14: Programa que pide dos números y nos da el resultado de su división.

Observa que hemos utilizado un operador de concatenación, lo que nos ha permitido emitir un mensaje formado por una frase seguida del resultado de una operación. Veremos estos operadores en breve.

OPERACIONES CON CADENAS Y NÚMEROS

Podemos realizar algunas operaciones con cadenas de caracteres y con números:

	Este bloque une las cadenas de caracteres que se coloquen en sus huecos. Si el contenido de los huecos son números o variables que contengan números, los junta como si fueran caracteres pero forman un número con el que se puede operar. Por ejemplo, al unir 2 y 4 el resultado es 24. Se pueden anidar varios para formar frases.
	Devuelve el carácter que ocupa el lugar indicado dentro de una cadena. Si dicho carácter es un número, lo trata como número y se puede operar con él.
	Devuelve el número de caracteres (incluidos los espacios) de la cadena o de la variable que contiene una cadena que se especifica.



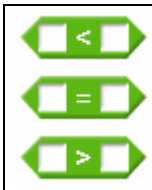
Ejemplo 15: Programa que te pregunta una palabra y te informa del número de caracteres que tiene y de la letra por la que empieza.

Otra forma con la que podríamos resolver el problema sería construyendo una lista con unos elementos que contienen las partes fijas (elementos 1 y 3) y otros que se van modificando en función de la palabra introducida (elementos 2 y 4). A esta lista le hemos llamado Frase.

OPERADORES RELACIONALES Y LÓGICOS

Scratch también dispone de **operadores relacionales** que nos devuelven el valor *true* (verdadero) o el valor *false* (falso) en función de que se cumpla o no la comparación que hace entre los valores indicados.

Los **operadores lógicos** Y y O combinan varias expresiones relacionales según se indica en la tabla. El operador lógico NO, invierte el valor lógico de una operación relacional.

	Devuelven el valor *true* (verdadero) si el dato situado a la izquierda es menor, igual o mayor, respectivamente que el dato situado a la derecha. Si los datos son números la comparación es la convencional. Si los datos son cadenas de caracteres, una cadena es menor que otra si alfabéticamente va antes. Por ejemplo: "blanco < tren" es verdadero, "justo > plato" es falso. No distingue mayúsculas de minúsculas.
	Operación lógica Y. Devuelve un valor *true* (verdadero) cuando las dos expresiones lógicas u operaciones relacionales colocadas en sus huecos son simultáneamente verdaderas. Si una o las dos expresiones son falsas, devuelve el valor *false* (falso).
	Operación lógica O. Devuelve un valor *true* (verdadero) cuando al menos una de las dos expresiones lógicas u operaciones relacionales colocadas en sus huecos es verdadera. Si ambas expresiones son falsas, devuelve el valor *false* (falso).
	Operación lógica NO. Devuelve el valor lógico contrario al que adopte la expresión lógica o relacional colocada en su hueco. Si la expresión es verdadera devuelve falso y viceversa.

14.- LA TOMA DE DECISIONES

Llegamos a una de las partes más importantes de la programación: la toma de decisiones. Existen unas instrucciones, llamadas "*sentencias condicionales*", que, tras evaluar si una *condición* determinada se cumple o no, deciden ejecutar un bloque de código o no ejecutarlo, o bien deciden ejecutar un bloque de código o ejecutar otro.

	Sentencia condicional **SI**. Ejecuta el conjunto de bloques de instrucciones que coloquemos en su interior únicamente si la condición que introduzcamos en el hueco es verdadera. La condición puede ser una operación relacional de comparación, o si se presiona una tecla o el ratón, o si un objeto toca a otro, etc.
	Sentencia condicional **SI ….. SI NO**. Si la condición que introduzcamos en el hueco es verdadera ejecuta el conjunto de bloques de instrucciones situado dentro de la abertura de SI, y si la condición es falsa, se ejecuta el conjunto de bloques situado en la abertura de SI NO.

Ejemplo 16: Vamos a diseñar un programa en el que el gatito nos pida dos valores numéricos. En caso de que el primero sea mayor que el segundo, realizará la resta. Si el primero es menor que el segundo, realizará la suma y si son iguales los multiplicará. Al final, el gatito dirá la operación realizada y su resultado.

Lo haremos creando una lista, llamada Frase, donde dejaremos dos huecos para rellenarlos con la operación y con el resultado.

15.- LA INTERACCIÓN ENTRE OBJETOS

Cuando empezamos ya dijimos que en un mismo programa de Scratch podemos tener varios objetos y todos ellos pueden interactuar entres sí y con el escenario. Veamos los más utilizados:

	Hace que el objeto en cuyo programa se ejecuta el bloque apunte al objeto seleccionado de la lista.
	Hace que el objeto en cuyo programa se ejecuta el bloque se desplace a donde está situado el objeto seleccionado de la lista.
	Hace que si el objeto en cuyo programa se ejecuta el bloque toca un borde del escenario rebote hacia atrás como lo haría una bola de billar al chocar con los bordes de la mesa.
	Devuelve *true* (verdadero) cuando el objeto en que se ejecuta el bloque toca al objeto seleccionado en la lista. Una de las opciones de la lista desplegable es “borde” (se refiere al borde del escenario).
	Devuelve *true* (verdadero) cuando el objeto en que se ejecuta el bloque toca un píxel del color seleccionado, que puede estar en otro objeto o en el escenario.
	Devuelve *true* (verdadero) cuando el primer color (situado dentro del objeto en que se ejecuta el bloque) toca un píxel del color seleccionado, que puede estar en otro objeto o en el escenario.
	Contiene el valor de la distancia en pasos del objeto en que se ejecuta el bloque al objeto seleccionado de la lista.
	Proporciona el valor de la propiedad o variable seleccionada de otro objeto que se selecciona (o del escenario (*stage*)).

Por otra parte, los objetos (incluido el escenario) pueden tanto enviar como recibir mensajes. Estos mensajes se pueden considerar como avisos y son utilizados por los programas para poner en marcha programas o detenerlos. Es más, como un mismo objeto puede tener varios programas, un mensaje enviado por un objeto puede estar dirigido a otro programa de ese mismo objeto.

	Se envía el mensaje seleccionado de la lista a todos los objetos y luego continua ejecutándose el siguiente bloque de instrucciones sin esperar a que se realicen las acciones de los programas activados.
	Se envía el mensaje seleccionado de la a todos los objetos y espera a que todos los programas activados por el mensaje terminen antes de continuar la ejecución del siguiente bloque de instrucciones.
	Ejecuta el programa que tiene debajo cuando recibe el mensaje seleccionado de la lista. Desde este bloque y desde los anteriores también se pueden crear nuevos mensajes.
	Con esta instrucción de cabecera, se puede iniciar un programa en un objeto cuando se produzca un cambio de fondo en el escenario al fondo seleccionado de la lista.

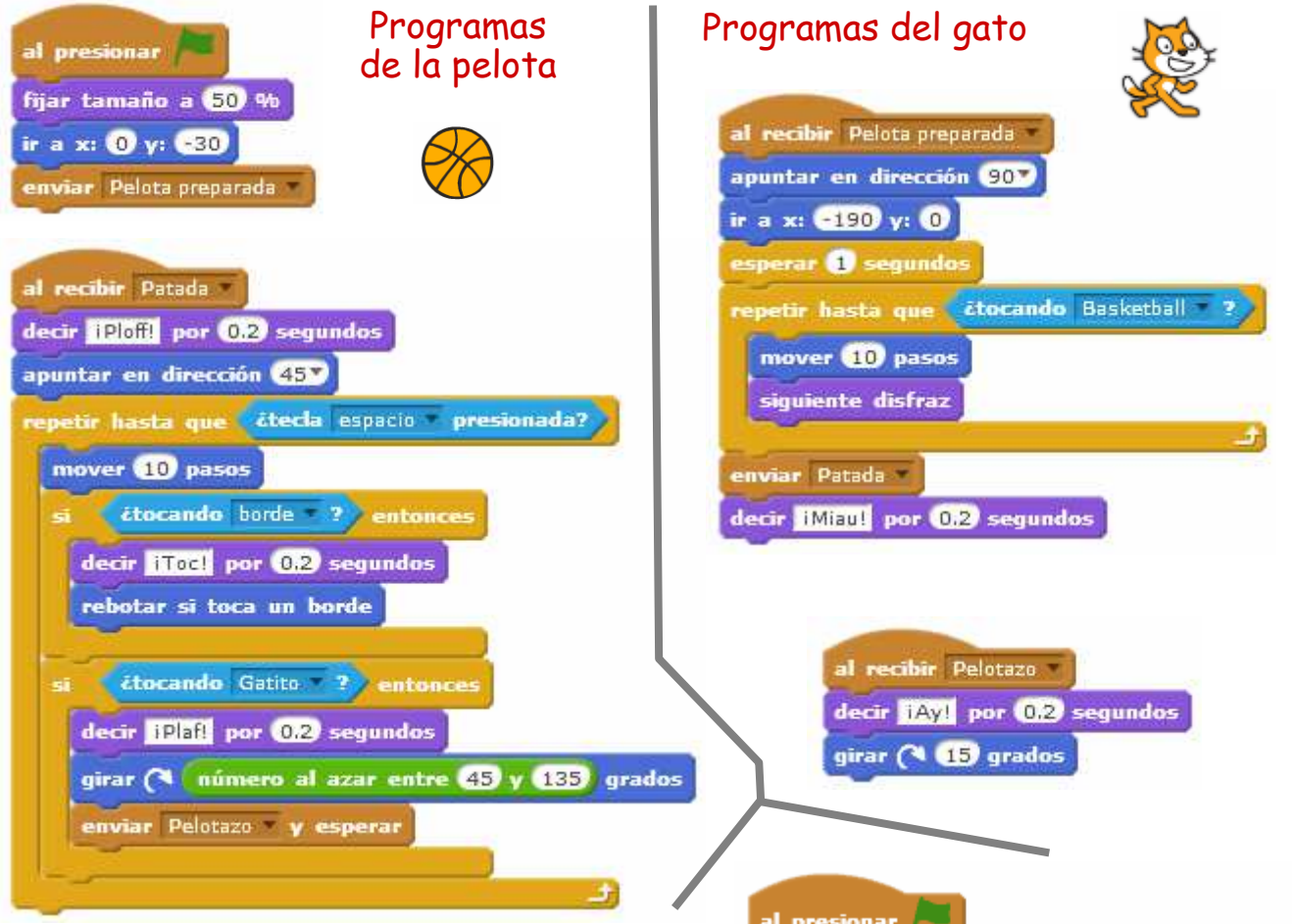
En un programa con varios objetos puede ocurrir que, al moverse, uno coincida encima de otro. En este caso ¿cuál tapa a cuál? Pues bien, cada objeto está en una capa y todas las capas están superpuestas unas sobre otras, pero podemos cambiar su orden. La capa que está más arriba tapa a las que están por debajo.

	Envía al objeto a la capa más encima de todas. Este objeto tapará a todos los demás hasta que ejecutemos la misma instrucción sobre otro objeto.
	Envía al objeto el número de capas hacia abajo que indiquemos. Si el número es negativo lo envía dicho número de capas hacia delante.

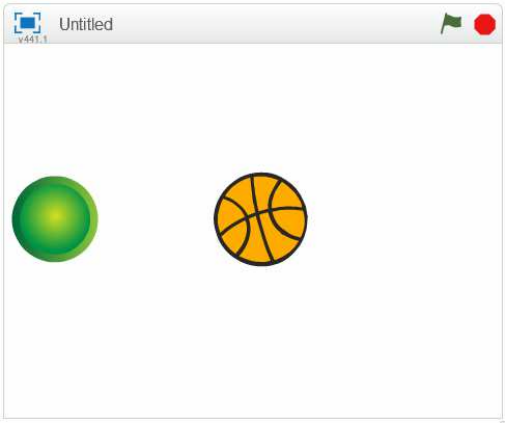
Ejemplo 17: Sacamos a escena el objeto Basketball  de la categoría Cosas de la biblioteca 

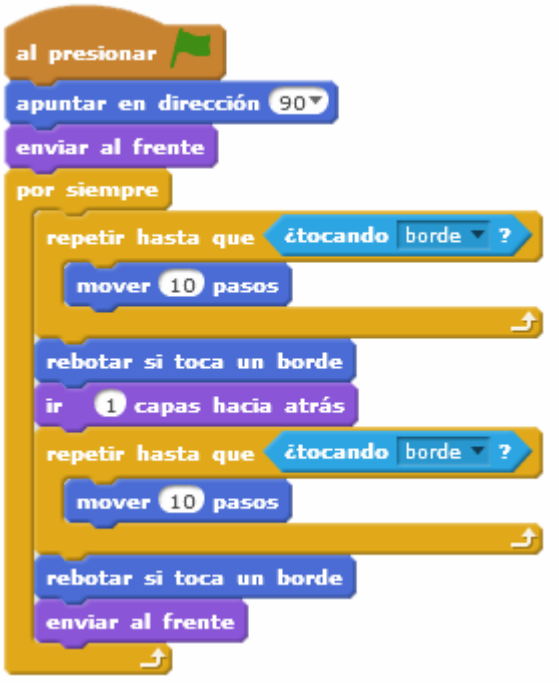
Haremos un programa que empieza situando al gato y a la pelota en sus posiciones de partida. A continuación el gatito avanzará cambiando de disfraz hasta tocar la pelota y emitiendo un ¡miau! Entonces la pelota, al recibir la patada, emitirá un mensaje de ¡ploff! y se pondrá en movimiento. Al chocar con un borde emitirá un mensaje ¡Toc! y rebotará y así sucesivamente hasta que toque al gato y se escuchará ¡Plaf! y el gato girará un poco del pelotazo y dirá ¡Ay! El programa terminará cuando presionemos la tecla espacio.

El programa saldría más simpático si emitiéramos sonidos en vez de mensajes, pero no disponemos de altavoces en los ordenadores del aula. En casa podéis hacerlo con sonidos.



Ejemplo 18: saca los objetos que se ven en la figura y colócalos como se indica. Construye y ejecuta el programa que se adjunta. Observarás que la pelota verde pasa unas veces por delante y otras por detrás de la pelota naranja. Lo conseguimos gracias a que vamos moviendo la pelota verde una capa hacia delante en un sentido y una capa hacia atrás en el contrario.





16.- LA DETENCIÓN DE LOS PROGRAMAS

Normalmente, la detención de un programa se produce al terminar de ejecutarse su conjunto de bloques de instrucciones, sin necesidad de incluir ningún bloque de detención del programa. No obstante, esto no es siempre lo que queremos. Existen instrucciones para detener el programa antes del final de su conjunto de bloques, por ejemplo, cuando se cumpla una condición o se presione una tecla, etc. También podemos detener programas desde instrucciones de detención ubicadas en otros programas del mismo objeto o incluso en programas de otros objetos diferentes.

	Detiene únicamente el programa en el que se ejecuta la instrucción.
	Detiene el resto de programas del objeto en que se ejecuta la instrucción, pero no el programa en el que está dicha instrucción.
	Detiene todos los programas, tanto del objeto en que se ejecuta la instrucción como de otros objetos.

Ejemplo 19: Abrimos Scratch y, además del gato, incluimos desde la biblioteca la pelota Basketball.

Si ejecutamos el programa, observamos que uno de los programas de la pelota lo que hace es que ésta permanezca permanentemente en movimiento y el otro que vaya cambiando de color. El programa del gato lo único que hace es hacer que el gato se mueve de izquierda a derecha constantemente. Experimentemos:

- Si presionamos la tecla “a” observamos que la pelota sigue moviéndose pero ya no cambia de color y el gato sigue moviéndose. O sea sólo se han detenido los otros programas del mismo objeto
- Si, a continuación, presionamos la tecla “espacio” (o desde el principio) se detienen todos los programas, tanto el movimiento de la pelota, como el cambio de color como el gato.
- Si la primera tecla que presionamos es la “b”, vemos que la pelota deja de moverse (ha detenido este programa) pero sigue cambiando de color y el gato moviéndose (no detiene otros programas del objeto ni los de otros objetos). Eso sí, como este programa está detenido, ahora, aunque presionemos la tecla “espacio” no se paran los otros programas pues ya el programa donde está esta instrucción está parado.

17.- LA INTERACCIÓN CON LOS USUARIOS

Hasta ahora, la única forma que hemos visto de interactuar el usuario con el programa ha sido mediante la introducción de números o cadenas de caracteres por teclado cuando nos los han pedido con el bloque "preguntar …. y esperar" y mediante la pulsación de una tecla con el bloque "¿tecla …. presionada?"

Sin embargo, existen otros bloques de instrucciones para que el usuario interactúe con el programa.

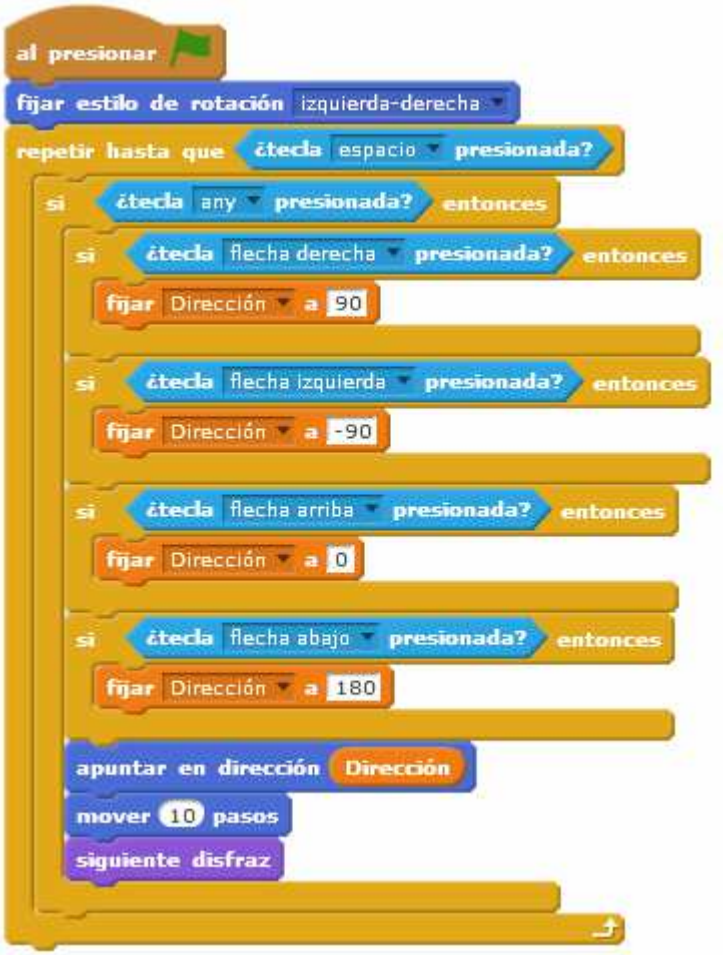
preguntar What's your name? y esperar	Formula una pregunta en la pantalla y guarda lo que introduzcamos en respuesta . Hace que el programa espere hasta presionar "Enter" o hacer clic en la casilla de verificación.
respuesta	Almacena la entrada por teclado más reciente tras la ejecución del bloque anterior por parte de cualquier objeto o del escenario. Es decir, es compartido por todos los objetos.
¿tecla espacio presionada?	Devuelve verdadero (*true*) cuando la tecla seleccionada está presionada. Se incluyen las teclas de flecha, es espacio, todas las letras (sin distinguir mayúsculas de minúsculas) y los números. También está la opción "*any*", que es cualquier tecla.
¿ratón presionado?	Devuelve verdadero (*true*) cuando el botón izquierdo del ratón está presionado.
posición x del ratón	Proporciona la coordenada X de la posición del puntero del ratón.
posición y del ratón	Proporciona la coordenada Y de la posición del puntero del ratón
volumen del sonido	Proporciona el volumen de los sonidos captados por el micrófono del ordenador en un rango de 1 a 100. Debemos conectar un micrófono.
apuntar hacia puntero del ratón	Apunta el objeto hacia la posición del puntero del ratón.
al presionar tecla espacio	Ejecuta el conjunto de bloques de instrucciones que tiene debajo al presionar la tecla seleccionada.
al hacer clic en este objeto	Ejecuta el conjunto de bloques de instrucciones que tiene debajo al hacer clic con el botón izquierdo del ratón sobre el objeto en cuyo programa se ejecuta la instrucción.

Ejemplo 20: Abre un nuevo programa y sitúa al gato en el centro de la pantalla aproximadamente. Construye y ejecuta el programa de la figura haciendo clic sobre el gato. Ahora mueve el ratón a izquierda y a derecha en la pantalla para que varíe su coordenada X.

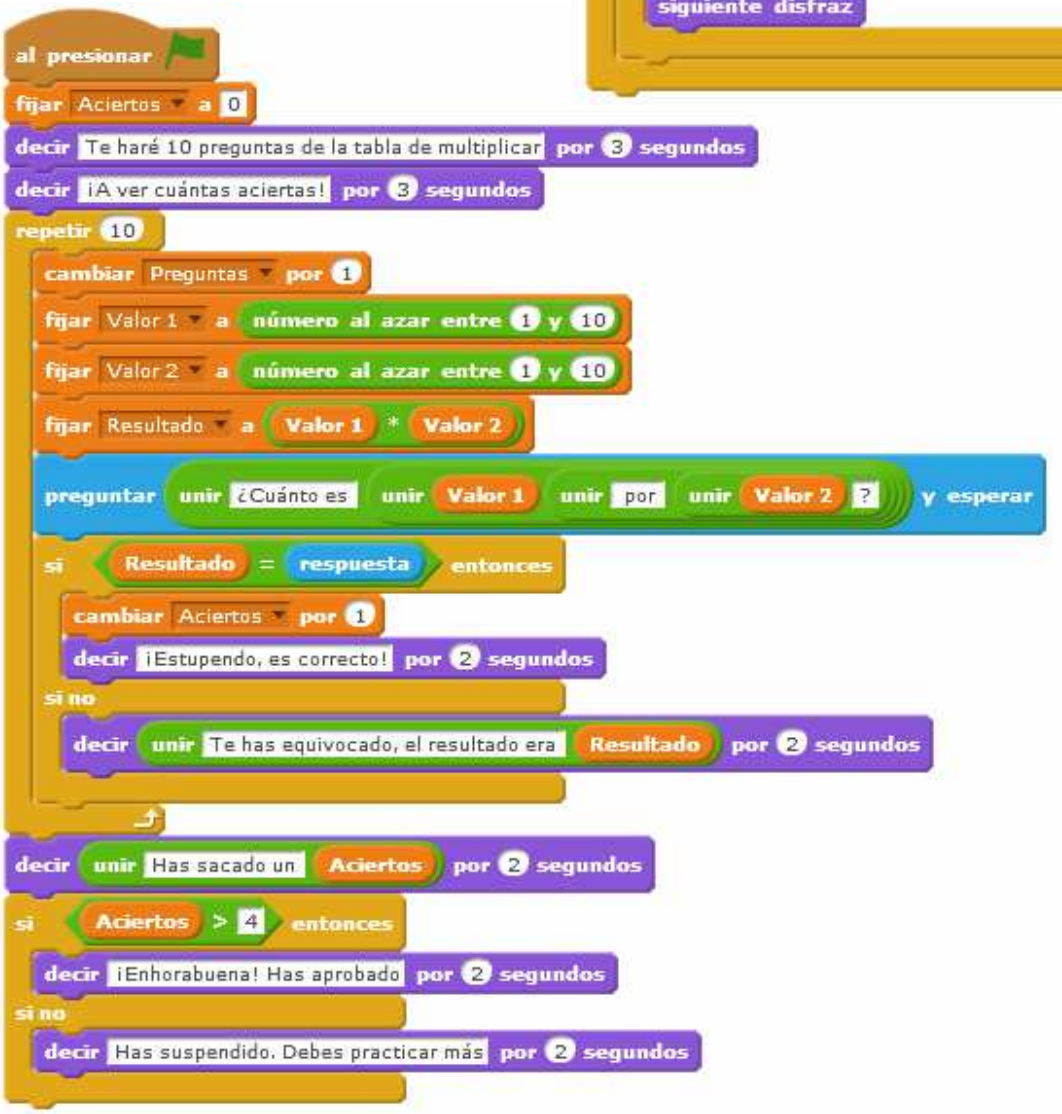
Observamos que cuando situamos el ratón hacia la derecha (coordenada X mayor de 30) el gato camina hacia la derecha; cuando situamos el ratón hacia la izquierda (coordenada X menor que –30), el gato camina hacia la izquierda; y cuando situamos el ratón por la parte central (coordenada X entre –30 y 30) el gato se queda parado.

Terminamos el programa haciendo clic con el ratón (hay que hacerlo fuera del gato pues de lo contrario empieza el programa de nuevo).

Ejemplo 21: Programa para poder mover al objeto por la pantalla con las teclas de flechas. El programa termina al pulsar la tecla "espacio".



Ejemplo 22: Programa para repaso de la tabla de multiplicar del 1 al 10. El programa hace diez preguntas de multiplicar con números al azar. Cuenta los aciertos y, por último, le dice al usuario los aciertos que ha tenido y si ha aprobado o suspendido.



Ejemplo 23: Juego de frontón con dos pelotas

En el juego intervienen los siguientes objetos:

- Raqueta (línea verde). La creamos. Tiene un movimiento vertical con las teclas de flecha arriba y abajo.
- Fondo (línea negra). La creamos. Está quieta; cuando una pelota la toca acaba el partido.
- Dos pelotas de la biblioteca de Scratch. Cada vez que chocan con la raqueta o con los bordes rebotan en una dirección al azar entre 45º y 135º. Cada choque aumenta en 1 el número de toques.

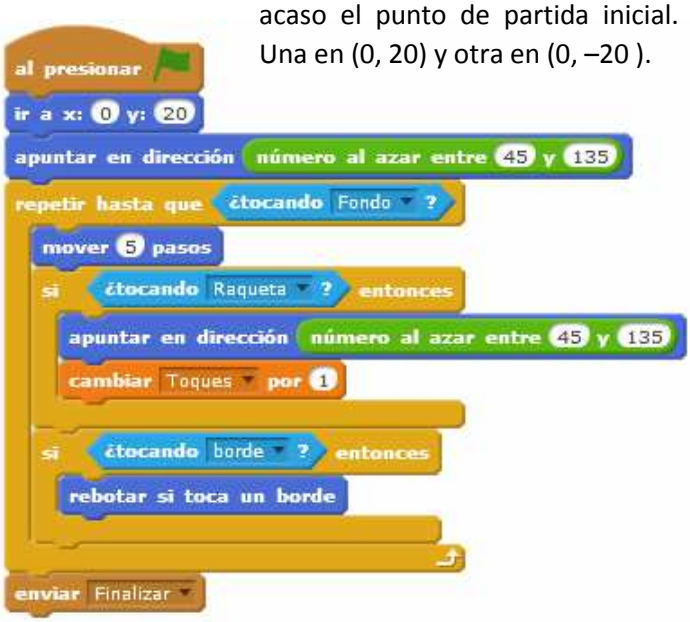
- Un niño que aparece al final para decir cuántos rebotes hemos conseguido dar con la raqueta.

Programas de la raqueta

Programas del niño

Programas de las pelotas

Ambas pelotas tendrían el mismo programa. Podemos modificar, si acaso el punto de partida inicial. Una en (0, 20) y otra en (0, –20).



Nota: podemos ajustar la velocidad de las pelotas y de la raqueta con el número de pasos de la instrucción mover. Cuanto menor, más lento será el movimiento.

18.- EL CONTROL DEL TIEMPO

En algunos programas podemos necesitar medir el tiempo que transcurre entre dos eventos determinados, o bien esperar un tiempo determinado desde que ocurre un evento para realizar otro. También podemos necesitar consultar la hora (hora, minutos y segundos) de nuestro ordenador, o bien la fecha (día del mes, mes y año) o el día de la semana (ojo, para scratch el primer día es el domingo).

Disponemos de las siguientes instrucciones:

cronómetro	El cronómetro es un contador en segundos (con una precisión de milésimas de segundo). Siempre está contando. Si marcamos la casilla que se sitúa a la izquierda en la paleta de bloques lo visualizamos en pantalla con una precisión de décimas de segundo.
reiniciar cronómetro	Reinicia a 0 el cronómetro al ejecutarse.
segundo actual	Proporciona el dato seleccionado en el menú de la fecha y hora del ordenador. El dato puede ser el año, mes, día del mes, día de la semana, hora, minuto y segundo.
días desde el 2000	Proporciona el número de días transcurridos desde el 1 de enero de 2000.

Ejemplo 24: Creamos un proyecto con los objetos de la biblioteca de Scratch que se indican en la figura:

La tarta tiene dos disfraces, uno con las velas encendidas y otro con las velas apagadas

El funcionamiento debe ser el siguiente: cada vez que hagamos clic con el ratón sobre la V, se encenderán las velas y cada vez que hagamos clic sobre la X se apagarán (imita al soplido de las velas). Ahora bien, si pasan 10 segundos sin que se haya soplado sobre las velas, éstas se apagarán solas.

Si una vez empezado a contar el tiempo de encendido se vuelve a pulsar la V, el tiempo empieza a contar de nuevo.

Este funcionamiento será permanente.

Programas de la tarta

Programa del botón V

Programa del botón X

19.- LA CREACIÓN DE BLOQUES PROPIOS (PROCEDIMIENTOS)

Scratch nos permite crear nuestros propios bloques. Incluso les podemos pasar parámetros de diferentes tipos (numéricos, de texto, etc) a dichos bloques.

La principal diferencia entre los bloques propios respecto a los que ya vienen definidos en la interface de Scratch es que los bloques propios se crean para un objeto específico, es decir, cada bloque propio sólo se puede usar en los programas del objeto en el que fue creado.

Los bloques propios son especialmente útiles en aquellos casos en que tenemos un conjunto de instrucciones que se repite en varias partes de los programas de un objeto de una aplicación. Podemos definir un bloque una sola vez y llamarlo cuantas veces queramos.

Para **crear un bloque** propio hacemos clic en "Más Bloques" de la paleta de bloques y, a continuación, en "Crear un bloque".

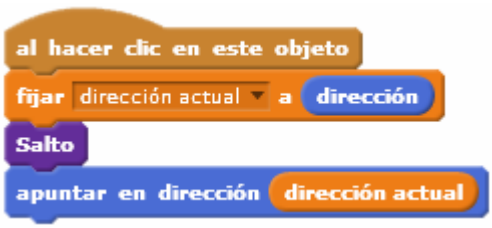
Nos pedirá que demos un nombre al bloque.

El siguiente paso depende de que nuestro bloque vaya a tener o no parámetros.

Creación de un bloque propio sin parámetros

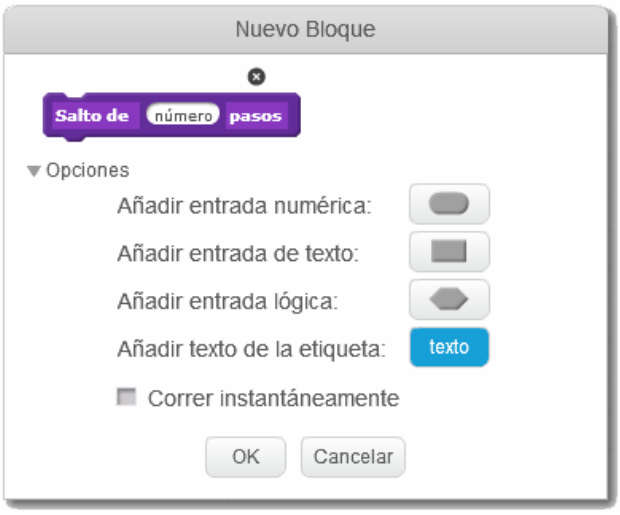
Le damos un nombre al bloque (por ejemplo," Salto") y hacemos clic sobre OK. Nos aparece el nuevo bloque en la paleta de bloques y una cabecera para definir el nuevo bloque propio en el área de programas.

Ejemplo 25: Creamos un proyecto con el objeto Basketball. La pelota estará desplazándose de forma permanente de izquierda a derecha y viceversa al rebotar en los bordes. Ahora bien, cada vez que hagamos clic sobre la pelota dará un salto hacia arriba y abajo y después seguirá moviéndose en la dirección que llevaba en ese momento.



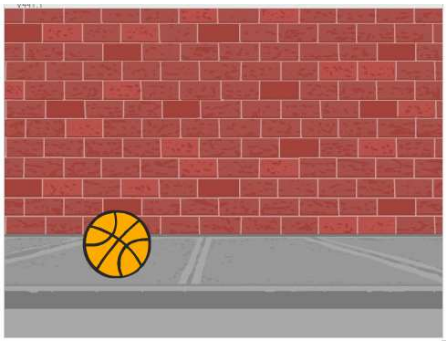
Creación de un bloque propio con parámetros

Le damos un nombre al bloque (por ejemplo," Salto") y hacemos clic sobre " ▸ Opciones". Nos aparece una lista de campos que podemos añadir (numérico, de texto, lógico y texto de etiqueta). Podemos añadir varios de cada tipo.

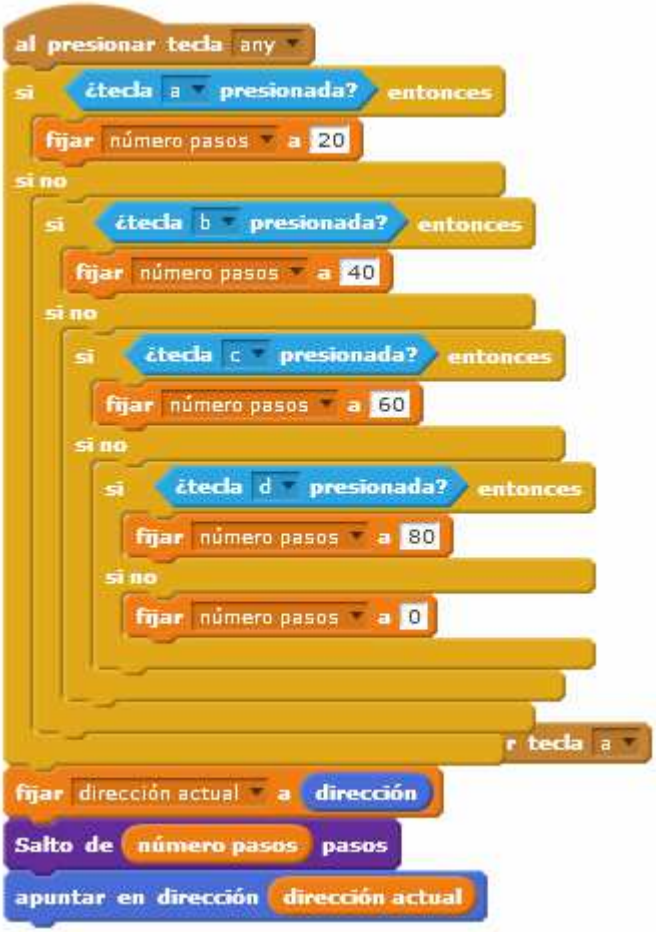


Si marcáramos la opción "Correr instantáneamente" lo que ocurre es que el bloque se ejecuta sin refresco de pantalla, es decir, no se refresca la posición del objeto hasta finalizar la ejecución del bloque (sólo vemos la posición inicial y final).

Tras hacer clic en OK, nos aparece el nuevo bloque en la paleta de bloques y una cabecera para definir el nuevo bloque propio en el área de programas.



Ejemplo 26: Creamos un proyecto con el objeto Basketball similar al anterior. La pelota tiene el mismo movimiento pero ahora la pelota saltará al pulsar teclas. Si pulsamos la tecla "a" el salto será de 20 pasos, si la "b" será de 40 pasos, si la "c" será de 60 pasos y si la "d" será de 80 pasos.



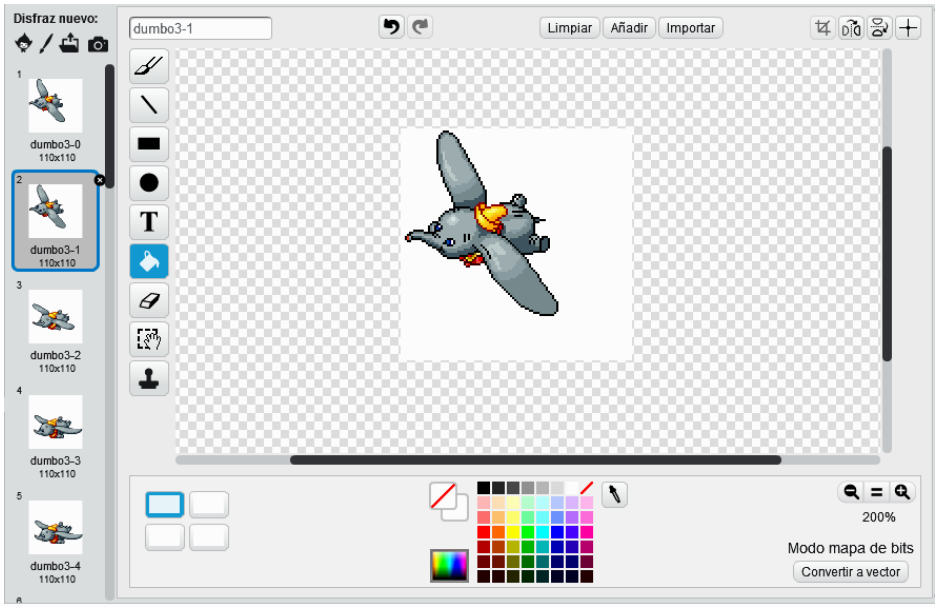
20.- TRABAJAR CON GIF ANIMADOS

Podemos importar GIF animados a nuestros programas Scratch. Al importar el archivo GIF animado, cada una de las imágenes del GIF se convierte en un disfraz del objeto.

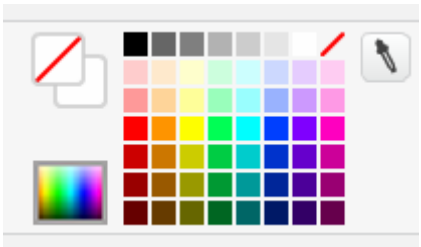
Para importar un GIF animado a Scratch actuaremos del modo siguiente:

- Primero lo descargamos desde Internet en una carpeta de nuestro ordenador.
- En la ventana "Nuevo objeto" de la interface de Scratch hacemos clic sobre el icono "Cargar objeto desde archivo". Navegamos hasta la carpeta donde tenemos el archivo y hacemos clic en abrir.
- Abrimos la solapa Disfraces y observamos que ya tenemos cada imagen del gif como un disfraz del objeto importado.

Normalmente las imágenes importadas como disfraces tienen fondo blanco. Estos disfraces pueden editarse para quitar el fondo blanco y hacerlo transparente. Para ello, seleccionamos el disfraz que queremos modificar y aparecerá en la ventana de edición de pintura.



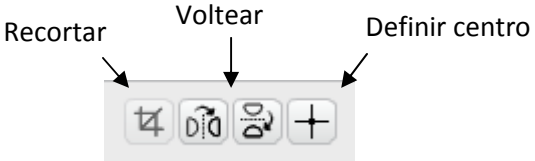
Seleccionamos el color transparente (cuadrito con línea roja diagonal)

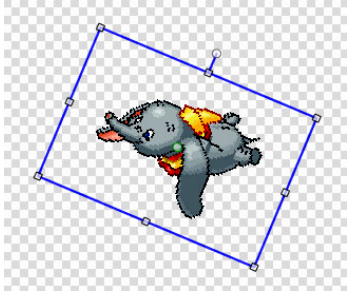


A continuación tomamos la herramienta "Rellenar con color" y hacemos clic sobre el fondo de la imagen.

Desaparece el fondo blanco y queda transparente

También es conveniente **definir el centro del disfraz**, lo cual es importante si el objeto debe ser girado en su posterior uso, ya que girará con respecto a este punto. También se puede **voltear** (en horizontal o vertical o **recortar**.





Para poder **girar** los objetos en la ventana de edición debemos primero seleccionarlos con el icono seleccionar y después girarlo con el ratón