1.3 SCRATCH 3

Scratch 3.0 es la última versión de la popular plataforma de programación visual creada por el grupo de investigación Lifelong Kindergarten del MIT. Con Scratch 3.0, puedes aprender a programar de manera divertida y creativa, sin necesidad de escribir código.

Esta versión ha sido diseñada para ser aún más accesible, potente y versátil, ofreciendo nuevas características y una interfaz más amigable para que cualquier persona, sin importar su edad o experiencia previa en programación, pueda dar vida a sus ideas.

En el siguiente capítulo de este e-book profundizaremos en el entorno de trabajo y uso práctico de esta versión de Scratch.

Antes de comenzar a programar con Scratch 3 es importante familiarizarse con algunos conceptos clave para aprovechar al máximo esta plataforma de programación visual.

A continuación conocerás la teoría relacionada a los conceptos y elementos que requieres para comenzar a trabajar con Scratch.

En el siguiente capítulo profundizaremos en estos conceptos desde la interfaz de usuario de Scratch en su versión 3.

1.3.1 Bloques y encaje de bloques

Los bloques son las piezas fundamentales de programación en Scratch. Cada uno representa una instrucción o acción específica, como moverse, cambiar disfraces y reproducir sonidos, entre otros.

Los bloques se pueden unir o encajar para crear secuencias lógicas y dar forma al comportamiento del proyecto (**Figuras 1.8. a 1.11**).

Los bloques son las piezas que se usan para crear el código en Scratch. Estos bloques, de colores y formas diversos, se conectan como las piezas de un puzle, formando lo que se conoce como script.

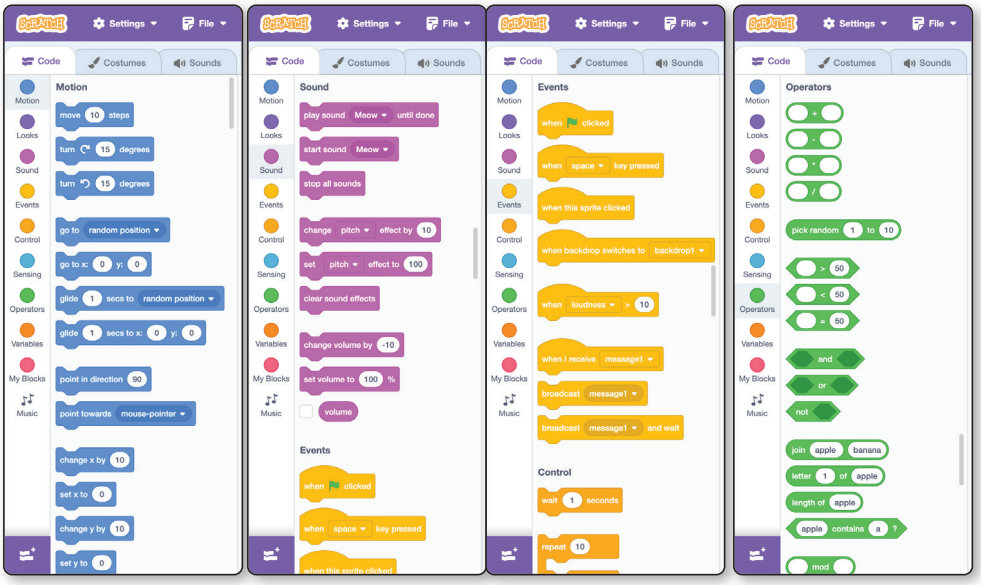


Figura 1.8. Movimiento. **Figura 1.9.** Sonido. **Figura 1.10.** Eventos. **Figura 1.11.** Operadores.

En Scratch los bloques y el **encaje de bloques** son fundamentales para crear programas visuales. Los bloques son piezas individuales de código que representan instrucciones específicas, como mover un objeto, cambiar el color de fondo o reproducir un sonido. Se encuentran en la paleta de bloques y están organizados en diferentes categorías según su funcionalidad, como **Movimiento**, **Apariencia**, **Sonido**, **Eventos** y **Control**, entre otros.

El encaje de bloques se refiere a la forma en que los bloques se conectan para formar una secuencia lógica de acciones.

Los bloques se encajan unos con otros como piezas de un rompecabezas para crear el flujo de ejecución del programa. Por ejemplo, si quieres hacer que un objeto se mueva hacia adelante y cambie de color al hacer clic en él, deberás usar bloques de movimiento y apariencia conectados en secuencia.

1.3.1.1 EJEMPLO: MOVIMIENTO DE UN OBJETO

Imagina que quieres que un personaje llamado **Gato** se mueva hacia la derecha al hacer clic en él; tendrías que usar los siguientes bloques:

- En la categoría **Eventos**: **Al hacer clic en Gato**–este bloque inicia el programa cuando se hace clic en el objeto **Gato**.
- En la categoría **Movimiento**: **Avanzar 10 pasos**–este bloque mueve el objeto Gato hacia adelante 10 pasos.

El encaje de bloques sería colocar el bloque **Al hacer clic en Gato** en la parte superior y luego conectar el bloque **Avanzar 10 pasos** debajo de él. De esta manera, cuando haces clic en el objeto **Gato**, se ejecutará el bloque de movimiento y el objeto se moverá hacia adelante.

1.3.2 Escenario y objetos

En Scratch el escenario y los objetos son elementos fundamentales para crear proyectos interactivos y juegos. El escenario es el fondo o la "pantalla" donde ocurren todas las acciones del proyecto, mientras que los objetos son los personajes o elementos visuales que interactúan en el escenario.

1.3.2.1 ESCENARIO

El escenario es como un lienzo en blanco donde puedes agregar fondos y crear mundos virtuales para tus proyectos.

Puedes cambiar el fondo del escenario para darle diferentes apariencias al proyecto y crear distintas escenas. Scratch incluye una biblioteca de fondos prediseñados y también permite importar tus propias imágenes para usar como fondos.

Imagina que estás creando un juego de carreras y quieres un escenario que simule una pista. Puedes seleccionar un fondo de pista de carreras de la biblioteca de Scratch o buscar una imagen en línea y cargarla como fondo. Luego, puedes agregar objetos de coches y otros elementos al escenario para completar la escena de la carrera.

1.3.2.2 OBJETOS

Los objetos son los personajes o elementos visuales que interactúan en el escenario. Cada objeto tiene su propio conjunto de bloques y puede realizar acciones independientes pero también podrán estar inter relacionados con otros objetos para lograr comportamientos más complejos o historias formadas con varias partes.

Los objetos pueden moverse, cambiar de apariencia, reproducir sonidos y responder a eventos específicos.

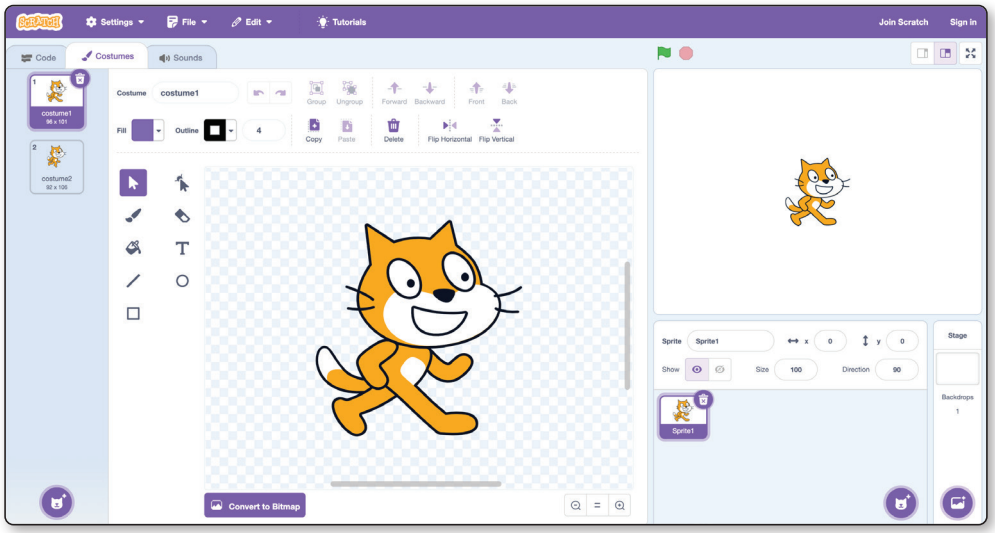
Siguiendo con el juego de carreras, puedes crear objetos que representen los vehículos que corren en la pista. Cada objeto de coche tendría su propio conjunto de bloques para moverse hacia adelante, girar, responder a comandos del teclado (por ejemplo, para acelerar o frenar) y detectar colisiones con otros objetos.

Además de los objetos del proyecto, Scratch también incluye el objeto **Escenario**, que representa el fondo de la pantalla.

Este objeto tiene su propio conjunto de bloques y puede realizar acciones globales que afectan a todo el proyecto, como cambiar el fondo, detener todos los objetos o activar eventos generales.

La combinación de escenario y objetos es lo que hace que los proyectos en Scratch cobren vida y sean interactivos.

Al diseñar tus escenarios y crear objetos con diferentes comportamientos, puedes dar rienda suelta a tu creatividad, y crear proyectos divertidos y educativos. (**Figura 1.12**).



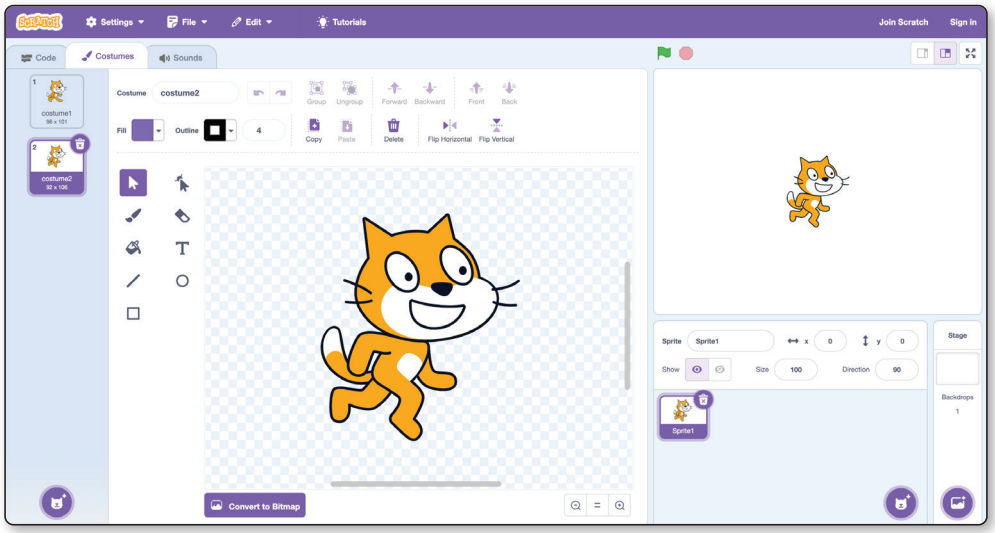


Figura 1.12. La posibilidad de programar interacciones entre los objetos y el escenario permite a los usuarios de Scratch crear juegos, historias, simulaciones y aplicaciones interactivas de una manera intuitiva y visual.

1.3.3 Eventos

En Scratch los eventos son bloques de programación que permiten a los objetos responder a acciones específicas, como clics de mouse, pulsaciones de teclas o interacciones con otros objetos. Son fundamentales para crear proyectos interactivos, ya que determinan cómo y cuándo los objetos deben ejecutar ciertas acciones.

Existen diferentes tipos de eventos en Scratch, algunos de los cuales son los siguientes:

- **Eventos de "Al hacer clic en bandera verde"**: este evento se activa automáticamente al hacer clic en la bandera verde que inicia el proyecto. Es útil para definir acciones iniciales que deben ocurrir cuando se inicia el proyecto.

  Imagina que estás creando un juego de memoria en el que el objetivo es emparejar cartas. Puedes utilizar el evento "**Al hacer clic en bandera verde**" para distribuir las cartas en posiciones aleatorias en el escenario cada vez que comienza el juego.

- **Eventos de teclado**: Scratch permite a los objetos responder a pulsaciones de teclas específicas. Puedes configurar eventos de teclado para que los objetos se muevan, cambien de apariencia o realicen cualquier acción deseada cuando se presionen ciertas teclas.

En el juego de carreras, puedes configurar un evento de teclado para que el objeto del coche se mueva hacia adelante cuando se presiona la tecla de flecha hacia arriba, y se detenga cuando se suelta la tecla.

- **Eventos de clic del mouse**: es posible configurar eventos para que los objetos respondan cuando se hace clic en ellos o cuando se hace clic en un área específica del escenario.

  En un proyecto de dibujo, puedes programar un evento de clic del mouse para que, al hacer clic en un color específico en la paleta de colores, el objeto del pincel cambie su color de trazo al seleccionado.

- **Eventos de colisiones**: los objetos pueden responder cuando colisionan con otros objetos en el escenario. Este tipo de evento es útil para crear juegos con interacciones entre objetos.

  En un juego de plataformas, puedes configurar un evento de colisión de modo que, cuando el personaje colisione con una plataforma, se quede en su lugar y no caiga.

- **Eventos personalizados**: Scratch permite crear eventos personalizados utilizando bloques de **Enviar** y **Cuando reciba**.

  En un juego de preguntas y respuestas, puedes configurar eventos personalizados de manera que, cuando el jugador responda correctamente a una pregunta, el objeto de texto muestre un mensaje de felicitación.

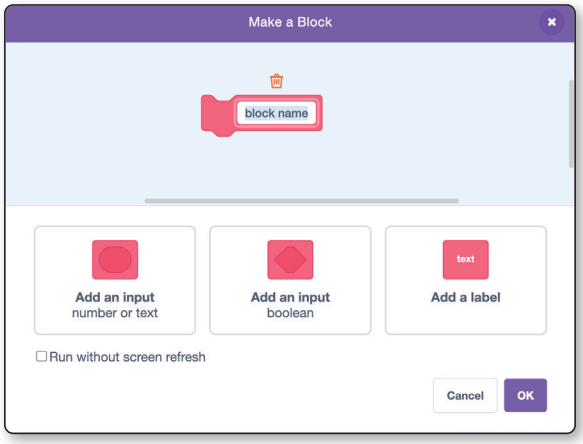


Figura 1.13. Los eventos en Scratch permiten crear interacciones y comportamientos complejos para los objetos y el escenario. Al combinar eventos con otros bloques de programación, como control, movimiento y apariencia, es posible crear proyectos más dinámicos y divertidos. La flexibilidad y la variedad de eventos en Scratch hacen que sea una herramienta poderosa para que los usuarios desarrollen su creatividad y aprendan programación de manera interactiva y visual.

1.3.4 Secuenciación

La **secuenciación** se refiere a la forma en que los bloques de programación se organizan en una secuencia lógica para que los objetos realicen acciones en un orden específico. Es fundamental para el **control de flujo** de ejecución de un proyecto y garantizar que las acciones ocurran en el momento adecuado. La secuenciación se logra mediante el encaje de bloques, donde los bloques se colocan uno después del otro en una secuencia lógica.

Por ejemplo, si quieres que un personaje se mueva hacia adelante y luego gire a la izquierda, primero arrastra el bloque de movimiento **Avanzar 10 pasos** y luego el bloque de movimiento **Girar en dirección a la izquierda 90 grados**. Al encajar estos bloques en orden, le dices al personaje que primero se mueva y luego gire.

La secuenciación es esencial para controlar el comportamiento de los objetos en Scratch y permite crear proyectos más complejos y avanzados. Aquí hay algunos ejemplos más de cómo se puede utilizar la secuenciación en Scratch:

- **Secuenciación de movimientos**: si quieres que un objeto se mueva en un patrón específico, puedes secuenciar varios bloques de movimiento uno después del otro para lograr el patrón deseado. Por ejemplo, puedes hacer que un objeto se mueva hacia adelante, gire, retroceda y luego gire otra vez en un bucle continuo.

- **Secuenciación de apariencia**: puedes secuenciar bloques de cambio de apariencia para crear animaciones. Por ejemplo, hacer que un objeto cambie su apariencia de manera gradual para simular una transición suave entre diferentes estados.

- **Secuenciación de eventos**: los eventos también se pueden secuenciar para crear interacciones más complejas. Puedes configurar una secuencia de eventos para que un objeto responda a diferentes acciones del usuario en un orden determinado.

- **Secuenciación de control**: al secuenciar bloques de control, puedes crear estructuras de decisión y bucles que permitan a los objetos tomar decisiones o repetir ciertas acciones en función de condiciones específicas.

- **Secuenciación de sonido**: puedes secuenciar bloques de sonido para crear efectos de sonido específicos que coincidan con las acciones de los objetos.

La secuenciación es una habilidad esencial en Scratch que permite a los usuarios controlar el flujo de ejecución de su proyecto y lograr comportamientos específicos en los objetos. Al organizar los bloques en una secuencia lógica, es posible crear proyectos más complejos y desafiantes, lo que mejora tu habilidad de programación y fomenta la creatividad en el proceso.

1.3.5 Control de flujo

El control de flujo en Scratch se refiere a la capacidad de tomar decisiones y repetir acciones en función de ciertas condiciones, con el fin de que los proyectos sean más interactivos y dinámicos. En Scratch el control de flujo se logra mediante el uso de bloques de control, que incluyen estructuras de decisión y bucles.

Estructuras de decisión (If-else): permiten hacer que un objeto tome decisiones basadas en ciertas condiciones. El bloque **si <condición> entonces** permite que un conjunto de bloques de código se ejecute solo si se cumple la condición especificada. Por otro lado, el bloque **sino** (**else**) permite especificar qué bloques se ejecutarán si la condición no se cumple.

Supón que quieres crear un juego en el que un personaje salte solo cuando se presiona la tecla de espacio. Puedes usar una estructura de decisión para lograrlo. El siguiente código muestra cómo se vería en Scratch:

```
Si <tecla espacio> pulsada entonces
 cambiar y <10> en y
sino
 cambiar y <-10> en y
```

En este ejemplo, el personaje cambiará su posición en el eje Y cuando se presione la tecla de espacio; de lo contrario, cambiará en la dirección opuesta.

Bucles (Repetir hasta, Repetir N veces): los bucles permiten repetir un conjunto de bloques de código varias veces hasta que se cumpla una cierta condición o hasta que se ejecute un número específico de veces.

Imagina que quieres hacer que un objeto se mueva de un lado de la pantalla al otro en un bucle continuo. Puedes usar un bucle **Repetir hasta** para lograrlo. El siguiente código muestra cómo se vería en Scratch:

```
Repetir hasta <tocar borde>
 cambiar x por <10>
```

En este ejemplo, el objeto se moverá 10 píxeles hacia la derecha en cada iteración del bucle, hasta que toque el borde de la pantalla.

Bucles anidados: también es posible combinar múltiples bucles para crear patrones más complejos y efectos visuales interesantes.

Si quieres hacer que un objeto se mueva en una forma circular mientras cambia de color en cada iteración, usa bucles anidados. El siguiente código muestra cómo se vería en Scratch:

```
Repetir 36 veces
 Cambiar color de fondo a <color>
 Repetir 10 veces
 Cambiar x por <10>
 Cambiar dirección a <10>
```

En este ejemplo, el objeto cambiará de color en cada iteración exterior del bucle, y se moverá 10 píxeles hacia adelante y 10 grados a la derecha en cada iteración interior.

El control de flujo en Scratch es una herramienta poderosa que permite crear proyectos más interactivos y complejos. Al utilizar estructuras de decisión y bucles, puedes controlar cómo se ejecutan los bloques de código y crear comportamientos más dinámicos en tus proyectos.

1.3.6 Variables

En Scratch las **variables** son elementos clave para almacenar y manipular datos durante la ejecución del programa. Las variables pueden contener diferentes tipos de información, como números, texto o valores booleanos (verdadero o falso). Al utilizar variables, puedes crear proyectos más interactivos y dinámicos, ya que es posible almacenar y cambiar información en tiempo real.

Creación y asignación de variables: para crear una variable en Scratch, simplemente haz clic en el bloque **Crear variable**, en la categoría **Variables**, y elige un nombre para darle. Luego, usa el bloque **Establecer <variable> a <valor>** para asignar un valor inicial a la variable.

Si deseas crear una variable llamada **puntuación** y establecerla inicialmente en cero, utiliza el siguiente código:

```
Al presionar bandera verde
 Crear variable “puntuación”
 Establecer puntuación a 0
```

Modificación de variables: una vez que se crea una variable, puedes cambiar su valor utilizando diferentes bloques, como **Cambiar <variable> por <valor>**.

Esto permite que las variables se actualicen y cambien a medida que el programa se ejecuta.

Continuando con el ejemplo anterior, para aumentar la puntuación del jugador en 10 puntos cada vez que toque un objeto, usa el siguiente código:

```
Cuando toques el objeto
 Cambiar puntuación por 10
```

Uso de variables en bloques de movimiento y apariencia: las variables también se pueden utilizar en bloques de movimiento y apariencia para crear efectos dinámicos y animaciones interesantes.

Si deseas crear un juego donde un personaje se mueva hacia la derecha y cambie de tamaño a medida que avanza, utiliza una variable para cambiar la posición en el eje X y el tamaño del personaje en cada iteración del bucle:

```
Repetir hasta <tocar borde>
 Cambiar x por (puntuación * 10)
 Cambiar tamaño por (puntuación / 10)
 Cambiar puntuación por 1
```

En este ejemplo, el personaje se moverá hacia la derecha y cambiará de tamaño en función de la puntuación actual, que aumenta en 1 en cada iteración del bucle.

Mostrar y ocultar variables: para visualizar el valor de una variable durante la ejecución del programa, utiliza el bloque **Mostrar variable** en la categoría **Pantalla**. Se mostrará el valor de la variable en la pantalla mientras se ejecuta el proyecto.

Si quieres mostrar la puntuación del jugador en la pantalla, puedes usar el siguiente código:

```
Al presionar bandera verde
 Crear variable “puntuación”
 Establecer puntuación a 0

Cuando toques el objeto
 Cambiar puntuación por 10
 Mostrar variable “puntuación”
```

En este ejemplo, cada vez que el jugador toque el objeto, la puntuación aumentará en 10 y se mostrará en la pantalla.

1.3.7 Sensibilidad a los contextos

La **sensibilidad a los contextos** es una característica importante de Scratch que permite a los bloques responder y reaccionar de manera diferente según el contexto en el que se encuentren. Esto significa que los mismos bloques pueden tener comportamientos diferentes dependiendo de dónde estén en el proyecto y de qué otros bloques los rodeen.

En Scratch, la sensibilidad a los contextos se basa en tres conceptos principales:

- **Conexiones de bloques**: los bloques en Scratch pueden conectarse entre sí para formar secuencias y estructuras lógicas. La forma en que lo hacen determina cómo interactúan y se comportan durante la ejecución del programa. Por ejemplo, un bloque **si-entonces** puede tener diferentes resultados dependiendo de qué bloque se coloque en su interior.

- **Bloques apilados**: cuando colocas un bloque encima de otro, se dice que están apilados. La sensibilidad a los contextos permite que los bloques apilados interactúen de manera diferente. Por ejemplo, si colocas un bloque **esperar 1 segundo** debajo de un bloque **mover 10 pasos**, el objeto primero se moverá y luego esperará, mientras que si los bloques estuvieran en el orden opuesto, el objeto esperaría antes de moverse.

- **Bloques en diferentes sombreados**: en Scratch los bloques de diferentes categorías están sombreados de distinta manera. Los bloques de control (como los bucles) tienen un sombreado más oscuro, mientras que los de movimiento, apariencia, eventos, etcétera, tienen sombreados más claros. Esta diferencia ayuda a identificar la jerarquía de los bloques y la manera en que interactúan entre sí.

  Supón que quieres crear un programa simple donde un personaje se mueva hacia la derecha y cambie de color cuando se haga clic en él. Utiliza la sensibilidad a los contextos para lograr este comportamiento.

  Para mover el personaje hacia la derecha, usa el bloque **mover 10 pasos** y conéctalo con el bloque **al presionar bandera verde**. Esto hará que el personaje se mueva hacia la derecha cuando comience el programa.

  Luego, para cambiar el color del personaje cuando se haga clic en él, utiliza el bloque **cambiar efecto color por 25** y conéctalo con el bloque **cuando hagas clic en este objeto**. Esto hará que el personaje cambie de color cuando se haga clic en él.

Si cambias el orden de los bloques, colocando **cambiar efecto color por 25** primero y **mover 10 pasos** después, el personaje cambiará de color cuando comience el programa y luego se moverá hacia la derecha cuando se haga clic en él. La sensibilidad a los contextos permite que los mismos bloques tengan comportamientos diferentes según su ubicación y conexión en el proyecto.

1.3.8 Compartir y explorar proyectos

Esta es una parte esencial de la experiencia de aprendizaje y creación. Los usuarios pueden **compartir proyectos** con la comunidad de Scratch para recibir comentarios, inspirar a otros y aprender de diferentes enfoques y técnicas. Además, explorar proyectos creados por otros es una excelente manera de obtener ideas, aprender nuevas habilidades y colaborar en proyectos en equipo.

1.3.8.1 COMPARTIR PROYECTOS

Para compartir un proyecto en Scratch, los usuarios deben seguir estos pasos:

- **Iniciar sesión**: es preciso tener una cuenta en Scratch y haber iniciado sesión en el sitio web.
- **Guardar el proyecto**: antes de compartir, asegúrate de haber guardado el proyecto. Haz clic en el botón **Guardar ahora** para conservar todos los cambios realizados.
- **Publicar el proyecto**: presiona en el botón **Compartir**, en la esquina superior derecha de la pantalla del editor. Se abrirá una ventana emergente que te permitirá dar información sobre el proyecto, como el título, una descripción, etiquetas y una imagen de portada. Completa los campos y pulsa en **Compartir** para publicar el proyecto en la comunidad de Scratch.
- **Explorar los proyectos compartidos**: una vez que hayas compartido tu proyecto, estará disponible en la sección **Explorar** del sitio web de Scratch, donde otros usuarios podrán verlo, probarlo y comentarlo. También puedes ver los proyectos compartidos por otros usuarios en esta sección.

1.3.8.2 EXPLORAR PROYECTOS

Para explorar proyectos creados por otros usuarios en Scratch sigue estos pasos:

- **Visita la sección Explorar en el sitio web de Scratch**: aquí encontrarás una amplia variedad de proyectos creados por usuarios de todo el mundo.
- **Filtrar proyectos**: aplica los filtros y las categorías disponibles para encontrar proyectos que te interesen. Puedes filtrar proyectos por tipo (juegos, animaciones, historias, etc.), nivel de dificultad, etiquetas y más.
- **Explora proyectos**: haz clic en un proyecto para abrirlo y ver cómo está hecho. Puedes ver y ejecutar el código del proyecto, probarlo y experimentar con él.
- **Comentar y remixar**: si encuentras un proyecto que te gusta, puedes dejar comentarios para el creador y expresar tu aprecio o hacer preguntas. Además, puedes hacer clic en el botón **Remixar** para crear una copia del proyecto y hacer tus propias modificaciones y mejoras.

1.3.9 Prueba y depuración

La fase de prueba y depuración es un paso crucial en el proceso de programación en Scratch. Durante esta etapa, los creadores de proyectos revisan minuciosamente su código para identificar y corregir errores o fallos que puedan afectar el funcionamiento y la experiencia del usuario. La prueba y depuración en Scratch se realiza a través de la observación, el análisis y la iteración para mejorar el proyecto y asegurarse de que funcione de manera óptima.

1.3.9.1 PRUEBA EN SCRATCH

La prueba en Scratch implica ejecutar el proyecto y evaluar su comportamiento y funcionalidad. Los creadores pueden probar su proyecto interactuando con él y siguiendo diferentes rutas posibles para asegurarse de que todas las funciones y características se ejecuten correctamente. Durante la prueba, es esencial estar atentos a los siguientes aspectos:

- **Funcionalidad**: verificar que todas las interacciones y eventos del proyecto funcionen como se esperaba. Asegurarse de que los bloques estén configurados correctamente y que los objetos respondan de manera adecuada a las acciones del usuario.

- **Lógica del programa**: revisar el código y asegurarse de que las secuencias de bloques estén en el orden correcto y que la lógica del programa sea coherente. Comprobar que los condicionales y bucles funcionen como corresponde.
- **Aspecto visual**: verificar que objetos, disfraces y fondos se muestren correctamente y que no haya superposiciones o errores de diseño.
- **Interacción**: probar la interfaz del usuario y comprobar que las instrucciones sean claras y fáciles de seguir.

1.3.9.2 DEPURACIÓN EN SCRATCH

La depuración en Scratch implica identificar y solucionar errores o problemas que se presenten durante la fase de prueba. Para hacerlo, los creadores deben:

- **Observar y analizar**: observar el comportamiento del proyecto y analizar el código para identificar posibles errores. Prestar atención a cualquier comportamiento inesperado o resultados incorrectos.
- **Utilizar la consola de depuración**: Scratch ofrece una consola de depuración que muestra mensajes específicos que pueden ayudar a identificar errores. Los creadores pueden utilizar bloques de **Mostrar** o **Decir** para imprimir mensajes en la consola y rastrear el flujo del programa.
- **Revisar bloques**: revisar minuciosamente cada bloque para asegurarse de que esté bien configurado y no haya errores tipográficos o de lógica.
- **Utilizar el bloque de pausa**: este bloque puede ser útil para detener temporalmente la ejecución del programa y observar el estado de los objetos y variables en un punto determinado.

1.4 ACTIVIDADES

A continuación se presentan las preguntas y los ejercicios que deberías saber responder y resolver para considerar aprendido el capítulo.

1.4.1 Test de autoevaluación

1. *¿Cuál es la característica principal de Scratch 2.0 en comparación con Scratch 1.0?*
2. *¿Qué son los bloques en Scratch?*
3. *¿Qué es un evento en Scratch?*
4. *¿Cuál es la función del bloque **Esperar**?*
5. *¿Cómo se utilizan las variables?*
6. *¿Qué es la sensibilidad al contexto en Scratch?*
7. *¿Cuál es el propósito de la fase de prueba?*
8. *¿Qué herramienta permite rastrear mensajes y errores durante la depuración?*

1.4.2 Ejercicios prácticos

1. *Dibuja tu nombre*

   Crea un proyecto que muestre tu nombre en el escenario utilizando bloques de movimiento para trazar cada letra. Puedes usar diferentes colores y efectos para hacerlo más interesante.

2. *Reproduce un sonido*

   Haz que al presionar una tecla específica, se reproduzca un sonido o una canción en el escenario. Puedes agregar varios sonidos y cambiarlos al pulsar diferentes teclas.

3. *Cambio de disfraces*

   Crea un personaje que cambie de apariencia al hacer clic sobre él. Utiliza bloques de disfraces para cambiar la apariencia del personaje cada vez que se hace clic.