Lección 20

Detección de colisión

Lección en línea
Ver en Code Studio

Propósito

Los estudiantes aprenden sobre la detección de colisión en la computadora. Trabajando en parejas, exploran cómo una computadora puede usar la ubicación de sprites y propiedades de tamaño y matemáticas para detectar si dos sprites se están tocando. Luego usan el bloque isTouching() para crear diferentes efectos cuando los sprites colisionan, incluidos los sonidos de reproducción. Por último, usan sus nuevas habilidades para mejorar el juego de desplazamiento lateral que comenzaron en la última Lección.

Esta Lección introduce formalmente el uso de abstracciones, formas simples de representar la complejidad subyacente.

En la última Lección, los estudiantes fueron expuestos a la idea de usar un bloque para representar código complejo. Los estudiantes exploran más a fondo esta idea en el contexto del desafío matemático complejo e intencional de determinar si dos sprites se están tocando. Al usar un solo bloque para representar esta complejidad, en este caso, el bloque isTouching, se vuelve mucho más fácil escribir y razonar sobre el código, y los estudiantes pueden apreciar el valor de usar abstracciones. En lecciones posteriores, los estudiantes continuarán construyendo sobre la abstracción isTouching() para crear interacciones de sprites más complejas.

Secuencia para el aprendizaje

Conocimiento inicial (10 min)

Ampliación del conocimiento

Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Detección de colisiones - contenido de la Lección
- Detección de colisiones - Guía de actividades

Vocabulario

- **Abstracción**: una representación simplificada de algo más complejo. Las abstracciones le permiten ocultar detalles para ayudarlo a manejar la complejidad, enfocarse en conceptos relevantes y razonar sobre problemas en un nivel superior.
- **Depuración**: encontrar y solucionar problemas en un algoritmo o programa.
- **If-Statement**: la estructura de programación común que implementa "declaraciones condicionales".

- Utilizar el bloque isTouching para determinar cuándo se tocan dos sprites.
- Describir cómo las abstracciones ayudan a manejar la complejidad del código.

Preparación

- Detección de colisiones - Guía de actividades de modo que cada par de estudiantes tenga una parte A y una parte B.

Código

- sprite.rotationSpeed
- sprite.velocityX
- sprite.velocityY

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Revisión: Recuerde brevemente a los estudiantes el juego de desplazamiento lateral que hicieron en la última Lección y pida que compartan cualquier idea que tengan para mejorar su juego en el futuro.

Pantalla: En un proyector o simplemente en una pantalla de computadora, muestre el juego que aparece en el primer nivel en Code Studio para esta lección. El maestro o un alumno puede controlar a la rana para saltar sobre un hongo. Asegúrese de que al menos algunos saltos sean exitosos y que algunos no vean que la detección de colisiones está funcionando.

Mostrar el juego: en este punto,sólo muestre el juego. Tienen una actvidad desconectada bastante importante inmediatamente después y importante superarla antes de interactuar con el juego.

Inducción: Una mejora interesante en este juego es que el hongo se mueve cuando la rana lo toca. ¿Puedes pensar en alguna forma en que la computadora pueda usar las propiedades de los sprites para averiguar si se están tocando entre sí?

Discute: Permita que los estudiantes hagan una lluvia de ideas sobre cómo la computadora podría determinar si los dos sprites se están tocando. Haga una lista de sus ideas en el pizarrón y dígales que tendrán la oportunidad de probar sus teorías en un momento.

Meta: El propósito de esta discusión es solo para obtener algunas ideas en la pizarra para que los estudiantes las utilicen en la próxima Lección. No es necesario evaluarlos o probarlos, porque los estudiantes trabajarán juntos para hacerlo inmediatamente después de la discusión

Ampliación del conocimiento

Colisiones desconectadas

Grupo: Agrupa a los estudiantes en parejas.

Observaciones: Ahora va a tener la oportunidad de probar las estrategias que surgieron como grupo. Cada guía de actividades tiene cuatro hojas de papel. Un compañero debe tomar los papeles con la “A” en la parte superior, y el otro debe tomar los papeles con la “B” en la parte superior. Cada uno dibujará dos sprites secretos en el gráfico, y su compañero tratará de averiguar si están tocando o no según la misma información que la computadora tendrá sobre las propiedades de cada uno de los sprites, así que no permita que su compañero mire lo que estás dibujando.

Distribuir: La Detección de colisiones - Guía de actividades para cada pareja. Asegúrese de que un compañero haya tomado las páginas con “A” en la parte superior y el otro haya tomado las páginas con “B” en la parte superior.

Cada alumno tendrá una línea para dibujar dos cuadrados. El estudiante elige la ubicación y el tamaño de cada uno de los cuadrados, y luego registra la información sobre los cuadrados en una tabla. A continuación, cambian las tablas (no los dibujos) y tratan de determinar si los dos sprites se están tocando en función del ancho de cada sprite y la distancia entre ellos.

La matemática para determinar si los sprites se están tocando es la siguiente:

- Resta las posiciones x (o y) de los sprites para encontrar la distancia entre sus centros.
- Divide el ancho (o la altura) de cada cuadrado por 2 para obtener la distancia desde el centro hasta el borde.
- Si la distancia entre los centros de los sprites es mayor que la suma de las distancias desde sus centros hasta sus bordes, los sprites no se tocan.
- Si la distancia entre los centros de los sprites es igual a la suma de las distancias desde sus centros hasta sus bordes, los sprites apenas se tocan.
- Si la distancia entre los centros de los sprites es mayor que la suma de las distancias desde sus centros hasta sus bordes, los sprites se superponen.

Recorra la sala: Apoye a los estudiantes mientras completan la hoja de trabajo. Si los estudiantes no están seguros de cómo determinar si los sprites se están tocando, aliéntelos a usar una de las ideas en el pizarrón. Recuerde que no están siendo calificados sobre si tienen razón o no, sino sobre su capacidad para usar el proceso de resolución de problemas. Si alguno de los estudiantes termina antes de tiempo, guíelos para que encuentren un método que funcione para sprites en cualquier lugar de la cuadrícula, no solo en la misma línea.

Compartir: Después de que todos los estudiantes hayan tenido la oportunidad de probar sus soluciones, pida que compartan lo que descubrieron.

Observaciones: La gente puede usar muchas estrategias diferentes para resolver un problema como éste. Debido a que las computadoras no pueden “ver” los dibujos de la misma manera que las personas, necesitan usar las matemáticas para determinar si dos cosas se están tocando. Veamos cómo esto puede funcionar a lo largo de una línea, pero podemos combinar estos métodos para trabajar en cualquier parte de la pantalla del juego.

IsTouching()

Transición: Haga que la clase inicie sesión en la Unidad 3- Niveles de Code Studio para la Lección 16.

Pantalla: Muestre la burbuja 2 y permita que los estudiantes observen y analicen por parejas cómo podrían codificar el comportamiento que ven en el programa.

Niveles de Code Studio

El propósito de este nivel es que los estudiantes vean que pueden probar si dos sprites se están tocando con los bloques que ya están disponibles para ellos. El código es complicado, pero solo usa bloques que los estudiantes ya han aprendido a detectar cuando dos sprites se están tocando.

Discuta: Pregunta a los estudiantes cómo se relaciona el código a continuación con lo que hicieron en la lección desconectada.

Observaciones: Aunque este código funciona, puede ser difícil de leer, y sería fácil cometer un error al escribirlo. También tomaría algo de tiempo escribir cada vez, por lo que un programador guardó el código en un bloque que podemos usar para determinar si dos sprites se están tocando, y no necesitamos escribir todo este código. Debido a que esto es algo que un programador quiere hacer una y otra vez, alguien ha creado un bloque llamado isTouching()que ya tiene todas las matemáticas dentro de él. Ocultar los detalles de cómo se hace algo para facilitar la programación se llama abstracción.

IsTouching()

Escribir las matemáticas cada vez que desees comprobar si dos sprites se están tocando puede llevar un tiempo, por lo que un programador creó el bloque isTouching, que puede comprobar si un sprite está tocando otro sprite (el objetivo). La computadora sigue haciendo los mismos cálculos que en el programa anterior, pero no tiene que preocuparse porque otro programador ya hizo ese trabajo.

Recorra la sala: Apoye a los estudiantes mientras trabajan en los niveles 3-8. El nivel 8 permite a los estudiantes trabajar en el desplazamiento lateral que crearon en la última lección. Es muy abierto, así que anima a los estudiantes a ser creativos y tratar de codificar las interacciones de los sprites que no han visto antes. Los estudiantes que completan el nivel temprano pueden continuar agregando nuevas interacciones o mejorando sus juegos de otras maneras.

Transferencia del conocimiento (5 min)

Preguntar:

¿Cuáles fueron algunas de las cosas diferentes que tus sprites hicieron cuando interactuaron?

¿Cuál es un tipo de interacción que te gustaría, pero que aún no has visto?

¿Tienes alguna idea de cómo podrías escribir el código para ese tipo de interacción?

Compartir: Permita a los estudiantes compartir el diferente tipo de interacciones que les gustaría ver. Hágales saber que aprenderán otros métodos para que los sprites interactúen más adelante.

Sugerencia para evaluación

Se sugiere los siguientes indicadores para evaluar formativamente los aprendizajes:

- Construyen y evalúan estrategias de manera colaborativa al resolver problemas no rutinarios.
- Incorporan el código, los medios y las bibliotecas existentes en los programas originales, para darles atribución.
- Prueban y perfeccionan sistemáticamente los programas.
- Documentan aquellos programas para que sean más fáciles de seguir, probar y depurar.