Universidad de Antioquia
Facultad de Ingenieria
Informatica II

Modelamiento proyecto final.

Integrantes:

Sebastian Bolivar Vanegas

Camilo Rojas Mendoza

Colombia, Medellín
2022

Índice general

Capítulo 1

Análisis del problema:

Crear un juego que pueda guardar y cargar datos, tenga implementación de 3 modelos físicos y cambio en la dificultad dependiendo del avance del jugador, generando así un entorno alejado de la monotonía en temas de dificultad, para imprimirle al video juego el factor de entretenimiento que destaque en nuestro juego para que el jugador no entre en aburrimiento.
La idea del proyecto es imprimirle creatividad y demostrar los conocimientos aprendidos y desarrollados durante el curso, lo cual será evidencia clara de las experiencias adquiridas durante el transcurso del curso.

Capítulo 2

Modelamiento del video juego:

El jugador avanza horizontalmente a través de un mapa con diversos obstáculos y enemigos intentando evitar su progreso. El jugador podrá saltar, deslizarse y lanzar explosivos para abrirse camino. Los enemigos podrán disparar y lanzar explosivos contra el jugador. También se podrá encontrar cajas obstaculizando el paso y trampas mortales que deberá esquivar usando los movimientos a su disposición. También si el jugador es muy adelantado por la cámara y sale del rango de visión, este perderá, lo que lo forzara a intentar minimizar el abuso de cierta mecánica.

2.1. Restricciones y condiciones.

- El juego debe tener implementados al menos 3 modelos de sistemas físicos.
- Debe poder guardar partidas.
- Dificultad creciente.

2.2. Dificultad.

La dificultad del juego irá incrementando en función de lo lejos sea capaz de llegar el jugador, habrán diferentes niveles con sus propias temáticas y cada nivel tendrá varios puntos de guardado donde el jugador tendrá un lugar seguro desde donde reiniciar al salir del juego.

2.3. Fisicas.

El juego tendrá 3 sistemas físicos principales implementados; rozamiento, MRU y tiro parabólico. También se intenta implementar MCU, cambio de medio y transmisión de energía.

El rozamiento se verá reflejado en una mecánica del movimiento del jugador al deslizarse para evitar algún obstáculo, esto reducirá su velocidad poco a poco. El MRU se verá en el movimiento normal del jugador. El tiro parabólico se apreciará en las mecánicas de disparo, tanto entre enemigos como del jugador. El MCU estará presente en algunos obstáculos rotatorios sobre un punto fijo. El cambio de medio afectará al movimiento del jugador, limitando su velocidad y aceleración normales. Finalmente la transmisión de energía se aplicará principalmente en los bloques móviles sobre el mapa.

Capítulo 3

Clases

El video juego se implementara con ayuda de las siguientes clases:

3.1. Manejo de archivos.

Esta clase se encargará de leer los datos de los archivos con los datos del juego y proveerlos al propio juego para que pueda hacer uso de estos.

Atributos:

- Nombre del archivo.
- Datos obtenidos del archivo.
- Datos para escribir al archivo (si no existía).

Métodos:

- Indicar el archivo a usar.
- Leer un archivo de texto.
- Escribir en un archivo de texto
- Cambiar datos.
- Datos a colocar al intentar abrir un archivo y no exista.
- Obtener datos.
- Sobreescribir, agregar o eliminar datos (En el archivo)

3.2. Ventana principal:

Esta clase estará enfocada en recibir al usuario para darle la opción de ingresar a su partida guardada en una cuenta o registrarse en una nueva, para

posteriormente enviarlo hacia la ventana de registro/ingreso.

Métodos:

- Botón de ingresar.
- Botón de registrar.

3.3. Ventana de registro/ingreso:

Aquí el usuario ingresa sus datos para ingresar a una cuenta pre existente o crear una nueva. La ventana verificará los datos que sean válidos y/o no estén repetidos. Al verificarse los datos de ingreso del usuario o registro de una nueva cuenta, se abrirá la ventana del juego con los datos de la respectiva cuenta.

Atributos:

- Variable de la clase de manejo de archivos.
- Variable que indica si es un inicio de sesión o registro.
- Variable que guarda los datos de todos los usuarios.

Métodos:

- Ingresar usuario.
- Registrar usuario.
- Verificar usuario.
- Botón de continuar.
- Botón de salir.

3.4. Ventana del juego

En esta clase el usuario puede ingresar a continuar el juego normal, un posible modo supervivencia y un apartado para cambiar su aspecto. Esta ventana controla los menús básicos del juego y el juego en sí, manejara los eventos tales como presionar una tecla, manejo de colisiones con un sistema de identificaciones para determinar el tipo de objeto que colisionó y llevar a cabo el proceso correspondiente.

Atributos:

- Datos del usuario.
- Velocidad de la cámara.
- Tipo de partida.

• Refresh rate
• Delta de tiempo

Métodos:

• Botones para recorrer los menús y opciones.
• Pausar juego.
• Detección de teclas pulsadas.
• Seleccionar aspecto del jugador.
• Guardar datos.
• Controlar puntaje.
• Crear escenario.
• Agregar actores.
• Calcular (diferncial)T.
• Muerte del jugador.

3.5. Jugador.

En esta clase tendran todos los atributos relacionados al jugador y sus métodos para interactuar con el juego.

Atributos:

• Posición.
• Velocidad máxima.
• Aceleración.
• Rozamiento experimentado.
• Fuerza de salto.
• Texturas.
• Salud.

Métodos:

• Moverse.
• Obtener valores relacionados al movimiento.
• Recibir daño.
• Morir.
• Cambiar texturas.

3.6. Enemigo.

En esta clase se tendrán los atributos y métodos de los enemigos definiendo su comportamiento. Atacará al jugador al entrar en un cierto rango.

Atributos:

- Posición.
- Velocidad.
- Textura.
- Objetivo.
- Arma.

Métodos:

- Moverse.
- Apuntar.
- Disparar.
- Recibir daño.
- Morir.

3.7. Checkpoint.

Esta clase es un objeto que cuando el jugador lo toca actualiza el punto de guardado y reinicio si sale y vuelve a entrar al juego, también regenera las vidas perdidas anteriormente.

Atributos:

- Número (Cuál de los checkpoints es)

Métodos:

- Regenerar jugador.
- Actualizar punto de guardado.

3.8. Plataforma.

En esta clase formarán los objetos que actuarán directamente relacionados al escenario del juego.

Atributos:

- Rompible.
- Fijo (Está fijo en el mapa).
- Tangible (Se puede interactuar con el).
- Posición.
- Velocidad.
- Aceleración.

- Punto de colisión.
- Energia del objeto.
- Texturas.
- Coeficientes de fricción.

Métodos:

- Moverse.
- Romperse.
- Girar.
- Cambiar texturas.
- Otorgar fricción.
- Obtener valores de colisión.

3.9. Obstáculo giratorio.

Atributos:

- Eje de giro.
- Velocidad angular
- Momento angular.
- Radio.
- Aceleración angular.
- Velocidad angular máxima.

Métodos:

- Actualizar.
- Es colisionado (Solo por explosivos del jugador).
- Actualizar valores del movimientor.

3.10. Generador de terreno.

Esta clase se encarga de entregar una lista de los bloques que formarán la parte sólida del mapa (no se pueden mover ni romper). Si está en modo supervivencia generará el terreno automáticamente empleando un generador de números pseudo-aleatorios para determinar la posición de cada bloque. Para evitar una generación caótica se darán pesos a los posibles resultados y se realizará una generación secuencial en la que cada bloque nuevo será posicionado a un lado del bloque anterior, el número aleatorio indicará en qué dirección será colocado.

Atributos:

- Datos de generación de niveles.
- Instancia del manejador de archivos.
- Texturas disponiblesr.
- Pesos de direcciones.
- Lista de bloques (Leídos o generados.

Métodos:

- Leer datos de archivo.
- Generar terreno (Usando las instrucciones de nivel).
- Generar terreno (Aleatorio).
- Generar siguiente bloque.
- Textura aleatoria.

3.11. Decorador.

Esta clase entregará la lista de obstáculos y bloques que el jugador puede romper o no puede tocar, también trabajará en conjunto con el generador de terreno cuando se necesite crear el terreno del modo supervivencia, su actuar será coordinado por la ventana del juego que pedirá al decorador generar bloques o una estructura aleatoria si y sólo si en ese lugar hay un bloque sólido de apoyo. Se intentará implementar un algoritmo de ruido de perlin para este propósito

Atributos:

- Datos de generación de estructuras.
- Instancia del manejador de archivos.
- Texturas disponiblesr.
- Lista de objetos (Leídos o generados).

Métodos:

- Leer datos de archivo.
- Generar terreno (Usando las instrucciones de nivel).
- Generar estructuras (Aleatorio).
- Ruido de perlin 2d (El valor indicará la estructura, entre más alto más bloques se generan y si supera cierto umbral se genera un obstáculo giratorio).
- Textura aleatoria.

3.12. Particulas.

En esta clase simplemente se generarán las partículas cuando una clase lo pida para agregar algunos efectos al juego.

Atributos:

- Posición.
- Cantidad.
- Duración.
- Textura.

Métodos:

- El constructor obtendrá los datos.
- Generar.
- Actualizar.
- Eliminar.

Capítulo 4

Procesamiento de datos

Los datos serán manejados por una clase para proveerlos al juego o guardarlos. Los datos se guardarán en un archivo de texto donde se podrán leer al iniciar el juego e intentar ingresar o crear una cuenta. Luego estos datos se pasarán a la ventana del juego para cargar el progreso de la cuenta del usuario. Al cerrar el juego se guardarán automáticamente los datos más recientes en la variable del programa y se actualizará el archivo de texto correspondiente. general.

Capítulo 5

Sprites y temas de diseño.