Tarea 7

Detalles de la tarea de esta unidad.

Enunciado.

ACTIVIDAD 1.

Con los conocimientos adquiridos durante la unidad vas a modificar un proyecto jME3 para incorporarle nuevas características.

El programa inicial ya realiza las siguientes tareas:

- **Crear una pared y un suelo.**
- **Activar el motor de físicas.**
- **Crear y empujar una bola en la dirección de la cámara.**

Tienes que implementar los siguientes elementos para superar la tarea:

a. **Colocar 10 bolos cilíndricos en forma de cuña que puedan caer de forma realista al ser golpeados.**
b. **Hacer que cada vez que se pulse la barra espaciadora o el botón izquierdo del ratón se lance una bola.**
c. **Poner un sonido ambiente que esté sonando durante toda la ejecución.**
d. **Emitir un sonido corto cada vez que se lance una bola.**
e. **Hacer que cuando se pulse la tecla G se active o desactive la gravedad del motor de físicas.**
f. **Hacer que se proyecten sombras de forma realista pero eficiente.**

También tendrás que entregar un documento en el que expliques los problemas que has encontrado y las soluciones que has adoptado durante la realización de la tarea.

NOTA: para los sonidos puedes usar los recursos que vienen incorporados en la librería jme3-test-data y que son accesibles desde "Sound/Effects/*" y "Sound/Ambient/*":

1- Crear bolos.

1.1-Método crearBolo()

Para crear los bolos, en primer lugar se ha creado cada bolo individualmente a partir del método crearBolo():

```
private void crearBolo(float x, float z) {
    Cylinder bolo = new Cylinder(16, 16, 0.2f, 1.0f, true);
    Geometry bolo_geo = new Geometry("bolo", bolo);
    bolo_geo.rotate((float) Math.toRadians(-90),0,0);
    Material bolo_mat = new Material(assetManager, "Common/MatDefs/Light/Lighting.j3md");
    bolo_geo.setMaterial(bolo_mat);
    bolo_mat.setBoolean("UseMaterialColors", true);
    bolo_mat.setColor("Ambient", ColorRGBA.Red);
    bolo_mat.setColor("Diffuse", ColorRGBA.Red);
    bolo_mat.setColor("Specular", ColorRGBA.White);
    bolo_mat.setFloat("Shininess", 1);
    bolo_geo.setLocalTranslation(x, 0.5f, z);
    rootNode.attachChild(bolo_geo);



    RigidBodyControl bolo_fis = new RigidBodyControl(1f);
    bolo_geo.addControl(bolo_fis);
    bulletAppState.getPhysicsSpace().add(bolo_fis);
}
```

Se crea un objeto **Cylinder** con:

- 16 segmentos en los ejes X y Y (para suavizar el cilindro).
- Radio de 0.2 unidades y altura de 1.0 unidad.
- Cierre en los extremos activado (true) para que tenga tapas.

Se genera una Geometry llamada "bolo", que utilizará esta forma cilíndrica.

El bolo se rota -90 grados en el eje X para que quede en posición vertical (por defecto, los cilindros en jME3 se crean acostados, ha sido una tarea complicada, ya que la opción Axis no se encuentra en esta versión y encontrar documentación actualizada es complicado).

Se ha creado un Material usando "Lighting.j3md", que permite iluminación realista, básicamente se han copiado los materiales de la bola pero modificando los parámetros y colores. Tras esto se ha asignado este material a la geometría del bolo.

Con bolo_geo.setLocalTranslation(x, 0.5f ,z): Se mueve el bolo a la posición (x, 0.5, z), ubicándolo sobre el suelo (asumiendo que el suelo está en y = 0).

Añadimos las físicas con un RigidBodyControl con masa 1.0, lo que hace que el bolo sea afectado por la gravedad y colisiones. Se agrega este control de física al bolo. Y se añade el bolo al espacio de física de bulletAppState, lo que permite que interactúe con otros objetos físicos.

1.2-Método crearBolos()

Para crear el conjunto de bolos se ha utilizado el método crearBolos(). Este método genera una formación de bolos en forma de cuña, donde la fila con 1 bolo está adelante y las filas con más bolos están más atrás.

```
private void crearBolos() {
    float startX = 0.0f, startZ = -4.0f;
    int filas = 4;

    for (int i = 0; i < filas; i++) {
        for (int j = 0; j <= i; j++) {
            float x = startX + j * 1.0f - (i * 0.5f);
            float z = startZ - i * 1.2f;
            crearBolo(x, z);
        }
    }
}
```

En primer lugar, se define la posición inicial y las variables de control:

- **startX = 0.0f**: Posición inicial en el eje X (centro de la formación).
- **startZ = -4.0f**: Representa la posición **más adelantada** de la formación.
- **filas = 4**: Define que habrá 4 filas de bolos en la formación de cuña.
- **El primer bucle i** recorre las filas, de 0 a filas (de 1 a 4 bolos por fila).
- **El segundo bucle j** coloca i + 1 bolos en cada fila, aumentando la cantidad en cada iteración.
- **float x = startX + j * 1.0f**: Espacia los bolos 1 unidad entre sí. **(i * 0.5f):** Centra cada fila para que se alineen correctamente:
- **float y= startZ - i * 1.2f:** Cada fila se coloca más atrás en el eje Z
- Llama a **crearBolo(x, z)** para:
    - Crear la geometría cilíndrica del bolo.
    - Aplicar materiales y colores.
    - Posicionar el bolo en (x, z).
    - Agregar física para permitir colisiones y caída realista.

2. Implementación de la opción para activar/desactivar la gravedad

Se ha añadido la posibilidad de activar o desactivar la gravedad en el mundo físico presionando la tecla G. Para ello:

a) Se ha agregado una variable de estado:

```
private Vector3f defaultGravity = new Vector3f(0, -9.81f, 0);
private boolean gravityEnabled = true;
```

- defaultGravity: Define la gravedad estándar del juego. Por defecto -9.81 m/s² que es la gravedad real de la Tierra.
- gravityEnabled: Indica si la gravedad está activada.

b) Se ha implementado el método para alternar la gravedad:

```
private void toggleGravedad() {
    gravityEnabled = !gravityEnabled;
    bulletAppState.getPhysicsSpace().setGravity(gravityEnabled ? defaultGravity : Vector3f.ZERO);
}
```

- Alterna entre la gravedad por defecto y Vector3f.ZERO (sin gravedad).

c) Se ha añadido la funcionalidad en los controles:

```
    private void configurarControles() {
    inputManager.addMapping("LanzarBola", new KeyTrigger(KeyInput.KEY_SPACE), new MouseButtonTrigger(0));
    inputManager.addMapping("ToggleGravedad", new KeyTrigger(KeyInput.KEY_G));
    inputManager.addListener(accionListener, "LanzarBola", "ToggleGravedad");
}
```

- ToggleGravedad se activa al presionar la tecla G.

3. Lanzamiento de bolas con efectos de sonido

Se ha añadido la funcionalidad para lanzar una bola al presionar la barra espaciadora (SPACE) o el botón izquierdo del ratón.

a) Configuración del control:

java

CopiarEditar

private void configurarControles() {

inputManager.addMapping("LanzarBola", new KeyTrigger(KeyInput.KEY_SPACE), new MouseButtonTrigger(0));

inputManager.addListener(accionListener, "LanzarBola");

}

- Se añade la acción LanzarBola.

a) Método lanzarBola()

Se ha sustituido el método crearBola() por el método lanzarBola(),

```
private void lanzarBola() {
    // Crear una esfera de 40 centímetros de diámetro
    Sphere esfera = new Sphere(32, 32, 0.4f);
    // Asociar la forma a una geometría nueva
    Geometry bola_geo = new Geometry("Bola", esfera);
    // asignarle el material
    bola_geo.setMaterial(bola_mat);
    // añadirla al grafo de escena
    rootNode.attachChild(bola_geo);
    // la colocamos en la posición de la cámara
    bola_geo.setLocalTranslation(cam.getLocation());
    //añadir las sombras realistas
    bola_geo.setShadowMode(RenderQueue.ShadowMode.CastAndReceive);
    // Creamos el objeto de control físico asociado a la bola con un peso
    // de 1Kg.
    RigidBodyControl bola_fis = new RigidBodyControl(1f);
    // Asociar la geometría de la bola al control físico
    bola_geo.addControl(bola_fis);
    // Añadirla al motor de física
    bulletAppState.getPhysicsSpace().add(bola_fis);
    // ¡Empujar la bola en la dirección que mira la cámara a una velocidad
    // de 8 metros por segundo!
    bola_fis.setLinearVelocity(cam.getDirection().mult(20));
    //añadir sonido al lanzar
    sonidoLanzar.play();
}
```

- **Velocidad**: Vector direccional de la cámara multiplicado por 20.
- **Físicas**: Masa 1kg para interacciones realistas con bolos.

b) Implementación de la acción:

```
private final ActionListener accionListener = (name, isPressed, tpf) -> {
    if (isPressed) {
        if (name.equals("LanzarBola")) {
            lanzarBola();
        } else if (name.equals("ToggleGravedad")) {
            toggleGravedad();
        }
    }
};
```

- Cuando LanzarBola es activado, se ejecuta el método lanzarBola().

c) Configuración del control

```
    private void configurarControles() {
    inputManager.addMapping("LanzarBola", new KeyTrigger(KeyInput.KEY_SPACE), new MouseButtonTrigger(0));
    inputManager.addMapping("ToggleGravedad", new KeyTrigger(KeyInput.KEY_G));
    inputManager.addListener(accionListener, "LanzarBola", "ToggleGravedad");
}
```

Se añade la acción LanzarBola al pulsar la barra espaciadora o el click derecho.

4. Añadir los efectos de sonido.

Se han implementado efectos de sonido de fondo y al lanzar la bola.

En primer lugar se han creado las variables de Audio:

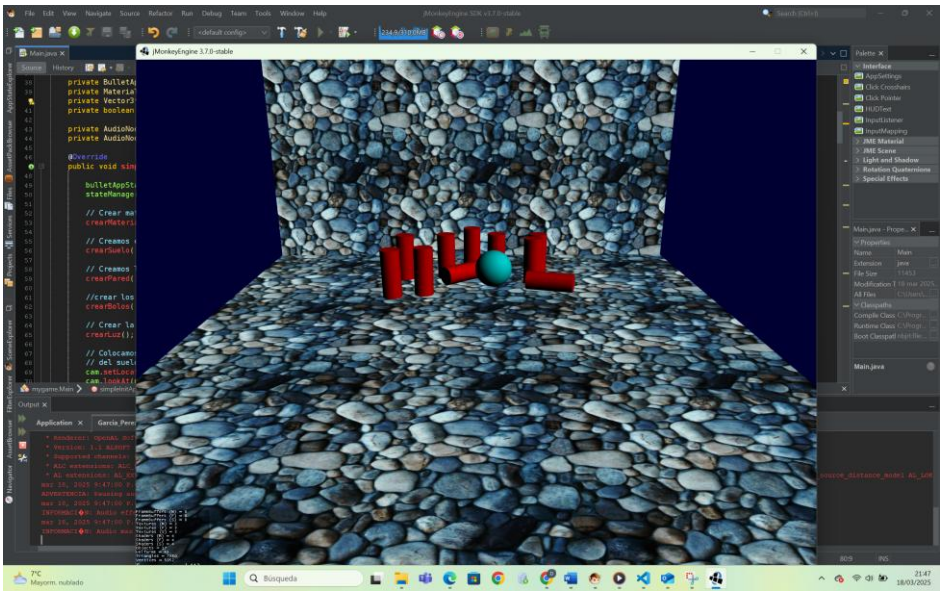
```
private AudioNode sonidoAmbiente;
private AudioNode sonidoLanzar;
```

a) Método iniciarSonidos

```
private void iniciarSonidos() {
    sonidoAmbiente = new AudioNode(assetManager, "Sound/Ambient/Nature.ogg", true);
    sonidoAmbiente.setLooping(true);
    sonidoAmbiente.setPositional(false);
    sonidoAmbiente.setVolume(0.5f);
    rootNode.attachChild(sonidoAmbiente);
    sonidoAmbiente.play();
    sonidoLanzar = new AudioNode(assetManager, "Sound/Effects/Sonido bola.wav", false);
    sonidoLanzar.setPositional(false);
    rootNode.attachChild(sonidoLanzar);
}
```

sonidoAmbiente: Se carga un sonido ambiental (Nature.ogg), que se reproduce en bucle.

sonidoLanzar: Se carga un sonido para cuando se lanza la bola (Sonido bola.wav).



ACTIVIDAD 2.

Creación de un entorno 3D The Jungle, como introducción al desarrollo de videojuegos 3D vamos a desarrollar un entorno como el descrito en la unidad, el cual incluirá únicamente una escena donde podremos pasear por el entorno.

Dicho entorno contendrá los siguientes pasos:

Paso 0: Realiza una pantalla inicial, con el título del juego, una imagen, un campo de texto y un botón para ir a la pantalla de juegos. En el campo de texto se pedirá el nombre de usuario que se pasará a la escena de juego.

Paso 1: Un **terreno** con al menos tres texturas aplicadas, el terreno tendrá variaciones en altura (colina, montaña, volcán…).

Paso 2: Incluiremos **vegetación** de forma espaciada y armónica a lo largo del terreno, incluiremos al menos dos tipos de árboles y dos de hierba.

Paso 3: Para mejorar la iluminación incluiremos un cielo (skybox) y luz ambiental (directional ligth).

Paso 4: Incluiremos un controlador de forma que nos podremos mover libremente por la escena para su exploración.

Paso 5: Inclusión de varios objetos 3D texturizados (ejemplo del cubo) a lo largo de la escena, entre 4 y 10 objetos iguales (ligeramente espaciados para que no haya que recorrer demasiado espacio).

Paso 6: Se mostrará en la parte superior de la pantalla un **interfaz de usuario** (UI) donde se indicará el número de objetos en el terreno , el nombre del jugador y una cuenta atrás de 60 segundos (el tiempo será configurable desde el inspector).

Si el personaje se acerca a un objeto y choca con el este desaparece (le recogemos), el interfaz reflejará el cambio en el número de objetos en el terreno y al llegar a cero se mostrará un mensaje indicando que se han recogido todos los objetos y si se desea volver a jugar.

Si el tiempo se acaba y no se han recogido todos los objetos, se mostrará un mensaje indicando que no se han conseguido recoger todos los objetos y si se desea volver a jugar.