Ejercicios de motores de físicas

Índice

1. Proyecto libgdx (*)

En las plantillas de la sesión tenemos dos proyectos relacionados: `ArmadilloScene2D` y `ArmadilloScene2D-android`. El primero de ellos es el proyecto libgdx Java estándar, mientras que el segundo es el proyecto que nos permite ejecutar el videojuego en Android.

a) Prueba a ejecutar la clase `ArmadilloScene2DDesktop` del proyecto `ArmadilloScene2D` como aplicación Java. Comprueba que el juego se ejecuta correctamente, y que puedes mover al armadillo por la pantalla utilizando en *pad* direccional en pantalla.

b) Ejecuta ahora el proyecto `ArmadilloScene2D-android` en un emulador o dispositivo Android (pon el emulador en horizontal).

2. Empaquetamiento de texturas (*)

Con libgdx se incluye una aplicación Java para el empaquetamiento de texturas con funcionalidades similares a la herramienta comercial TexturePacker, pero de forma gratuita. Puedes encontrar esta herramienta en el proyecto `TexturePacker` incluido en las plantillas de la sesión.

a) En el proyecto tenemos un programa principal de ejemplo llamado `EspecialistaTexturePacker`. En el código vemos que debemos especificar el directorio que contiene las imágenes de entrada, y el directorio donde se generará la textura resultante. Prueba a ejecutar el programa con distintos formatos de textura y observa el resultado.

b) Hemos comentado que libgdx también soporta el formato TMX para los fondos de tipo mosaico, pero no se puede utilizar directamente, sino que hay que procesar previamente estos ficheros. Esto lo podemos hacer con la herramienta `TiledMapPacker` incluida también en este proyecto. Tenemos un programa principal de ejemplo que la utiliza (`EspecialistaTileMapPacker`). Prueba a ejecutarlo y observa el resultado obtenido.

3. Motor de físicas

Tenemos en las plantillas un par de proyectos que contienen un juego en libgdx que utiliza el motor de físicas Box2D: `LatasBox2D` y `LatasBox2D-android`. El juego consiste en lanzar una bola para derribar una pila de latas. La bola se lanza pinchando sobre ella (según la posición en la que pinchemos se lanzará con una determinada velocidad y dirección). Por el momento lo único que aparecerá en el escenario es dicha bola. Se pide:

a) En la clase `Box2DFactory` tenemos los métodos que se encargan de crear los cuerpos del mundo 2D. Introduce en el método `createBounds` el código necesario para crear los límites del escenario. Comprueba ahora que al lanzar la bola no se sale de la pantalla.

b) En el método `createCan` de la misma clase anterior, introduce el código necesario para definir un cuerpo dinámico en las coordenadas *(x,y)* proporcionadas, y defínelo como una caja de tamaño *width x height* de densidad 1. Comprueba que tras hacer esto el juego funciona correctamente.

4. Detección de contactos

Vamos a añadir al juego anterior detección de contactos para añadir puntuación cada vez que una lata sea golpeada por la bola u otra de las latas. Esto lo definiremos en los métodos de `ContactListener` incluidos en la clase `Simulation`. Utilizaremos dos enfoques distintos:

a) Incrementar el daño en función del impulso calculado en `postSolve`.

b) Comentar el código anterior, y tener en cuenta sólo el momento en el que se inicia el contacto para calcular la puntuación. Para ello introduciremos en `beginContact` el código necesario que calcule la velocidad del impacto.