Tecnología de Videojuegos
Práctica 5: Bucle principal de un juego

UAH, Departamento de Automática, ATC-SOL
`http://atc1.aut.uah.es`

Objetivos:

- Implementar el bucle principal de un videojuego
- Diseñar una pequeña jerarquía de objetos
- Implementar la lógica princpal de un videojuego
- Utilizar métodos y atributos estáticos

Comentario inicial

El objetivo de esta práctica es implementar el bucle principal de un videojuego. Para ello se ampliará la jerarquía de clases desarrollada hasta el momento para introducir nuevas clases, incluyendo una clase que realice la visualización en modo texto del mapa del juego. Finalmente se implementará el bucle princpal del videojuego.

El resultado final será un RPG rudimentario en modo texto que visualiza un mapa en forma de matriz de 20x20. Dentro de esa matriz se situará el jugador con un personaje de su elección, así como una serie de enemigos y monedas de oro, plata o bronce con distinta puntuación. El objetivo del juego es obtener la máxima puntuación recogiendo las monedas, para lo que el jugador tendrá que desplazarse por la cuadrícula evadiento a los enemigos. Si enemigos y jugador coinciden lucharán siguiendo el esquema desarrollado en prácticas anteriores.

Ejercicio 1

Expanda la clase `Personaje` como se detalla a continuación:

1. Añada dos campos privados `private int x` y `private int y` que contendrán la posición del personaje dentro del mapa.
2. Implemente un método `public void setPosicion(int x, int y)`.
3. Implemente métodos que permitan el movimiento del personaje: `moverArriba()`, `moverAbajo()`, `moverDerecha()` y `moverIzquierda()`, que desplazan una posición en el sentido indicado. Suponga que el mapa es toroidal y que sus dimensiones vienen dadas por la clase `Mapa`, implementada en el ejercicio 3.
4. Añada un nuevo campo `private int puntuacion` que contendrá la puntuación del personaje. Introduzca los métodos asociados `public void aumentarPuntuacion(int puntos)` y `public int getPuntuacion()`. Inicialmente la puntuación del personaje será cero.

Ejercicio 2

Diseñe e implemente una nueva jerarquía de clases, dentro del paquete `p3.monedas`, que represente a las monedas dispersas en el mapa. Las monedas pueden ser de oro, plata y bronce, tendrán una puntuación asociada al tipo de moneda (4, 2 y 1 respectivamente) que se tiene que poder consultar externamente, y contendrán métodos que permitan ubicar las monedas dentro del mapa. Consulte con el profesor del laboratorio si tiene dudas sobre cómo implementar la jerarquía.

Ejercicio 3

Implemente una nueva clase llamada `Mapa`, dentro del paquete `p3`. Esta clase se encargará de visualizar el mapa del juego. Para ello contendrá dos atributos estáticos públicos para indicar el número de columnas y filas del mapa, y un único método (también estático) `visualizarMapa`, al que se pasará como argumentos una lista de las monedas dispersas en el mapa, otra lista con los enemigos y un objeto que representa al propio personaje del juego.

¿Por qué se definen tanto los campos como el método de la clase `Mapa` como estáticos? ¿qué ventajas aporta?

Sugerencia: Utilice código desarrollado en prácticas anteriores para representar el mapa.

Ejercicio 4

Modifique la clase `Juego` desarrollada en la práctica anterior para contener el bucle principal del videojuego, que viene detallado a continuación:

1. Preguntar al usuario su nombre y qué tipo de personaje quiere jugar.
2. Crear el persona.
3. Crear aleatoriamente enemigos y monedas[1]
4. Visualizar el mapa.
5. Pedir al jugador un movimiento (arriba, abajo, izquierda o derecha)
6. Implementar la acción del jugador (mover personaje, detectar colisiones, etc)
7. Mover aleatoriamente a los enemigos (mirar clase `Random` en Internet)
8. Detectar colisiones de enemigos con el jugador.
9. Visualizar el mapa.
10. Volver a 5.

En caso de que el jugador se mueva a donde hay una moneda, se añade la puntuación correspondiente, y se borra la moneda del mapa. En caso de que el jugador coincida con un enemigo se inicia un combate.

Para modularizar mejor el el código, introduzca la lógica anterior dentro de los métodos `private void inicializarJuego()` y `private void actualizarJuego()`.

[1] Mirar clase `Random` en Internet.