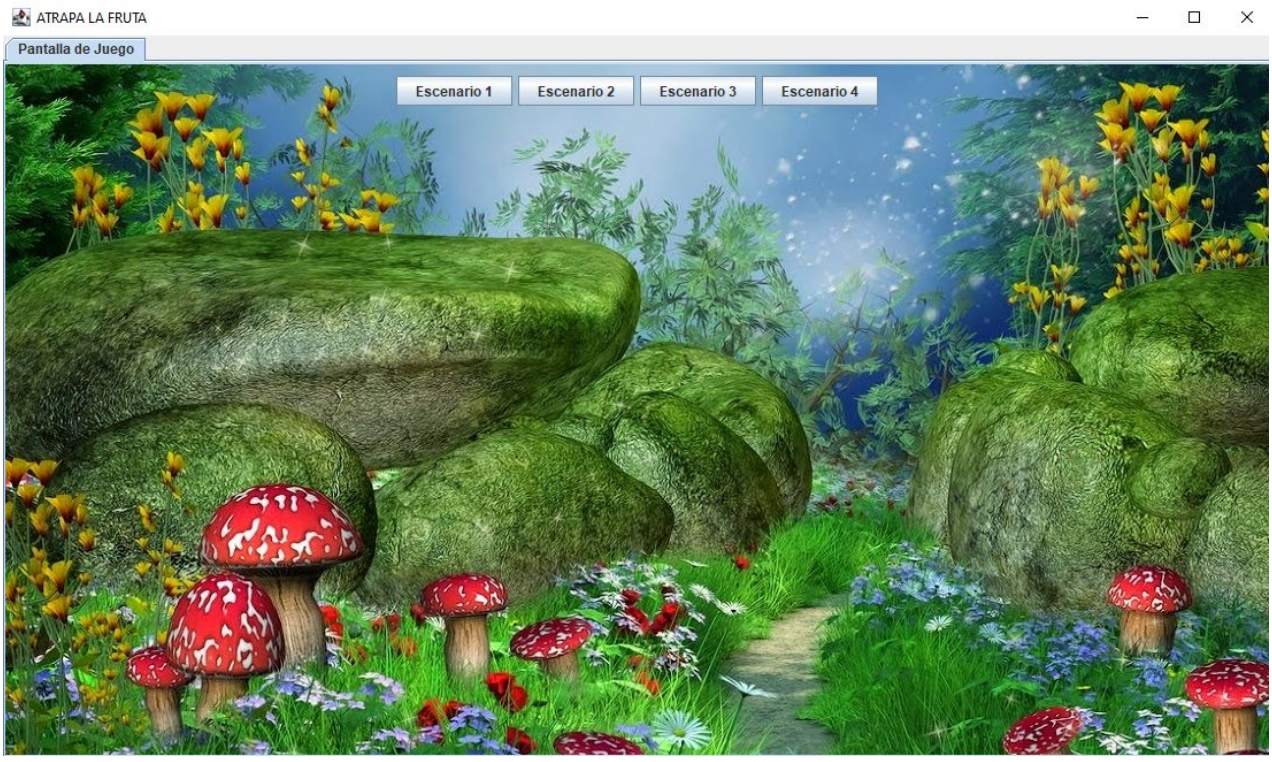
Universidad Politécnica de Cartagena

ATRAPA LA FRUTA



Datos del proyecto:

-Autor: Rubén Pedreño Martínez

-Nombre del proyecto: ATRAPA LA FRUTA

-Profesor de prácticas adjunto: Daniel Pérez Berenguer

-Profesor de teoría adjunto: Pedro Sánchez Palma

-Entorno de programación: Java

-IDE empleado: Eclipse

SECCIÓN CONSOLA

Durante esta sección, el código se encuentra orientado a la salida de información en pantalla por consola. El proyecto consta de 9 clases, cada una de ellas con una función particular:

Clase "TimerControl"

Esta clase tiene dos objetivos principales. El primero es almacenar el objeto "ventana" de la clase "Ventana", con el fin de ser solicitado por otras clases, así como para dibujar el escenario de forma sencilla. El segundo es servir como temporizador que guie los procesos temporales del programa.

Esta clase hereda de "TimerTask", de forma que implementa el método "run", método donde dibujaremos el escenario cada vez que ocurra un ciclo del temporizador "Timer"(siempre que se cumpla la condición de seguir jugando y no ocurra una excepción del tipo "java.lang.ArrayIndexOutOfBoundsException").

En caso de que la condición de seguir jugando sea falsa, el programa termina. En caso de que ocurra la excepción previamente mencionada (la cual ocurrirá antes de que el programa termine),se llamará al método "finPartida" para que realice los últimos ajustes de la simulación.

Clase "Ventana"

Esta es la clase central del programa, y la que contiene los métodos "más importantes" (aunque sin una de las 9 clases, nada funcionaría). Almacena el objeto de tipo Escenario.

Esta clase contiene el método "dibujarEscenario", que permite dibujar el escenario con cada iteración del Timer, haciendo uso de las variables de tamaño del escenario y un vector de elementos que contiene las diversas frutas y personajes a dibujar. Para dibujar el escenario simplemente dibuja "#" en consola simulando el suelo. También contiene el método "seguirJugando", que establece las condiciones mínimas del programa para que no finalice su ejecución.

El método "check", que gestiona los eventos de avanzar el personaje, que este salte o que coja una fruta (ayudándose del método "check_desenfoque" para gestionar el periodo de invisibilidad del personaje una vez atrapa una fruta HarryPoter).

El método "finPartida" que llama al método asociado para mostrar las frutas y recorre todos los elementos, haciendo uso de sus métodos particulares para mostrar datos.

Finalmente, algunos getters y setters.

Clase "Escenario"

El escenario se encarga de almacenar los elementos que serán dibujados en este. Esta clase está orientada a servir como selector de escenario, de forma que es su funcionalidad principal.

Tiene un método principal, denominado "seleccionEscenario", el cual, en base a un número previamente seleccionado por consola, este crea uno de los 4 posibles escenarios y su ventana asociada, así como los elementos que se van a mostrar en este. Dado que el método va a ser llamado por el Timer, aprovecho para que devuelva la ventana y pueda ser almacenada.

Para crear los elementos, este método se apoya en los métodos "agregarFruta" y "agregarPersonaje", los cuales, dada una posición particular del vector de elementos y los atributos del elemento en particular, crean el objeto y lo almacenan.

Finalmente contiene métodos getter para vectores de frutas o de personajes (y un getter en particular para el protagonista).

Clase "Elemento"

Se trata de una clase abstracta de la que heredarán las frutas y personajes. La utilidad de esta clase es almacenar los atributos comunes de los diversos elementos: posiciones "x" e "y" en consola, ancho y alto de los elementos (no será necesaria en esta sección), imagen del elemento (no será necesaria en esta sección) y símbolo del elemento (con que letra será representado por consola).

También contiene un getter del símbolo como ayuda al método "dibujarEscenario" e inicializa el método "mostrarDatos", que posteriormente sobrescribirán las clases que hereden de esta.

Clase "Personaje"

Hereda de Elemento y añade algunos atributos como el nombre y la velocidad, así como la sobreescritura del método "mostrarDatos".

Esta clase contienen la mayoría de los métodos a los que el método "check" en "Ventana" hace referencia. Un método avanzar que varía la posición "x" del personaje en consola, un método saltar que varía la posición "y" se apoya en la variable "chech_salto" para saber cuando debe de caer al suelo de nuevo.

Un método "cogerFruta" que gestiona el tipo de fruta que se ha cogido y llama al efecto particular de dicha fruta, así como stackear los puntos actuales y los dados por esta fruta. También hace desaparecer la fruta de consola haciendo que su símbolo sea un espacio en blanco.

Finalmente el método "mostrarFrutas", que muestra todas las frutas obtenidas y algunos de sus datos como el nombre y la puntuación otorgada. También devuelve la puntuación total.

Clase "Fruta"

Hereda de elemento y actúa a su vez como superclase de "FrutaMágica". Añade los atributos extras "nombre" y "puntos", así como la sobreescritura del método "mostrarDatos".

Clase "FrutaMágica"

Esta clase hereda de "Fruta" y añade como atributos los 3 efectos posibles de las frutas. Contiene un método "aplicarEfectos", que gestiona el efecto que se debe aplicar al personaje en base del efecto que provoque la fruta cogida.

Se apoya en los métodos "desenfocar"(hace al protagonista invisible siguiendo la misma metodología que para eliminar frutas del escenario una vez cogidas),"correr"(aumenta la velocidad con la que el personaje se desplaza, si es posible y no hay restricción de velocidad) y "ralentizar" (reduce la velocidad con la que el personaje se desplaza, siempre que este no se quede parado).

Finalmente, el método "mostrarDatos" sobrescrito, como en el resto de las clases que heredan.

Clase "SalidaPantalla"

Contiene los String's asociados a la presentación del proyecto y la selección del escenario por pantalla.

Clase “Main”

Y finalmente la clase principal. En esta clase creo un objeto “cronómetro” de tipo “Timer” para que gestione los eventos temporales del programa.

Creo un objeto de tipo TimerControl (hereda de TimerTask) y le paso como argumento la ventana para que la almacene. Para conseguir la ventana, aprovecho y llamo al método “dibujarEscenario” (que me devuelve al mismo tiempo la ventana) con un escenario auxiliar que posteriormente hago nulo.

Tras esto establezco la tarea del TimerControl y le paso como argumento el Timer y el tiempo que debe esperar el cronómetro hasta recibir un aviso.

Finalmente doy el valor por defecto “Start” a “lectura” y me introduzco en un bucle “while” infinito, donde constantemente recojo información por consola y la almaceno en la variable “lectura”.

SECCIÓN SWING

En este caso, el código del proyecto se verá orientado hacia una salida por pantalla mediante Swing, aunque buena parte del código se reutiliza.

Las clases empleadas serán las mismas, a excepción de la clase "SalidaPantalla"(que no nos resulta útil debido a que ya no hay salida por consola) y la clase "TimerControl" (puesto que en esta sección emplearemos el Timer que nos proporciona Swing).

Clase "Main"

La clase principal cambia completamente con respecto a la vista en la sección de consola. En este caso hereda de JFrame, de forma que crearemos un objeto de tipo "JFrame" con el nombre "ATRAPA LA FRUTA", que hará la función de ventana de nuestro programa.

Permitiremos que esta se pueda cerrar con el "Exit on close" y crearemos un "JTabbedPane", que permitirá tener varios JPanel`s separados por pestañas en caso de que lo necesitásemos.

Creo el objeto Ventana que heredará de JPanel y le paso el JTabbed para almacenarlo y usarlo posteriormente.

Tras esto creo una pestaña del JTabbed y en ella asocio el JPanel "Ventana" que acabo de crear.

Finalmente introduzco el JTabbed en el paño de contención, lo empaqueto y lo hago visible.

Como último detalle, muestro un JOptionPane con la presentación del juego.

Clase "Ventana"

Si en la sección de consola era una clase importante, ahora lo es incluso más, ya que realiza prácticamente todas las acciones del juego. Cambia mucho con respecto a la clase vista en la sección de consola.

En su constructor creo un JPanel asociado a los botones y le doy un tamaño adecuado que se ajuste a la posición de estos. Tras esto hago el JPanel transparente, para que no afecte a la imagen de fondo posterior.

Tras esto establezco el tamaño deseado del JPanel y creo un objeto de tipo escenario. Inicializo la imagen de fondo de presentación y creo un KeyListener que actuará como Listener de las teclas del teclado, avisando de cualquier estímulo al DirectionListener.

En la siguiente sección inicializo todos los botones de tipo JButton y les asocio un ActionListener para gestionar los estímulos generados por estos.

Almaceno todos los botones en su JPanel asociado y tras esto guardo dicho JPanel en el principal.

Finalmente creo un objeto Timer y le asocio el tiempo entre impulsos, así como un listener de tipo Rebound Listener.

La siguiente sección del código está conformada por las tres clases privadas asociadas a los listeners de los botones, el Timer y el teclado respectivamente:

-Para los botones, compruebo que botón ha sido pulsado y en base a ello, cambio la imagen inicial de presentación por la imagen asociada al escenario seleccionado. Tras esto hago invisible el JPanel de los botones para que no moleste durante la partida, permito que se

comiencen a pintar los elementos del escenario, inicio el Timer y hago un “repaint” para actualizar el estado del JPanel principal.

-Para el Timer, compruebo si sigo jugando. Si no es el caso, detengo el Timer y llamo al método “limpiarEscenario” para que elimine todos los elementos presentes.

En caso de que siga jugando, llamo al mismo método “check” que en la sección de consola y hago un “repaint” para actualizar el estado del JPanel.

-Para el teclado, compruebo si se ha pulsado la tecla “w” de salto y me aseguro de que el valor del contador sea mayor que 49 pulsaciones, para que el personaje no pueda saltar indefinidamente y se mantenga en el aire flotando si mantenemos la tecla pulsada.

Si este tiempo de cooldown ha pasado, reseteo el contador y hago que el protagonista salte.

Tras esto pasamos al método paintComponent, el cual es llamado cada vez que ocurre un “repaint”. En este método, dibujo la imagen de fondo del JPanel, compruebo si puedo pintar elementos y recorro el vector de elementos pintándolos todos.

Aplico un “try and catch” para la línea de pintar elementos, de forma que, si uno de los elementos no puede ser pintado porque ha salido de la ventana (lo cual solo puede ocurrir si el personaje ha llegado al final), muestro por pantalla las frutas recogidas durante la simulación y algunos de sus datos mediante un JOptionPane y termino el programa.

Los métodos “seguirJugando” y “check” son prácticamente iguales a los de la sección por consola, aunque los métodos auxiliares en los que se apoya si que han cambiado con respecto a los anteriores:

-En check_desenfoque, seguimos la misma mecánica, aunque para hacer invisible al personaje, le cambiamos su “ImageIcon” por un png transparente, de forma que de el efecto de invisibilidad.

-En check_choque cambiamos la mentalidad totalmente y, en lugar de comprobar si las posiciones del personaje y las frutas coinciden, creamos una caja de colisión para cada elemento, de forma que, si las cajas de colisiones impactan, significa que el personaje ha cogido una fruta.

-En check_salto, seguimos la misma filosofía que para la sección de consola.

Finalmente un par de getters

Clase “Escenario”

La clase escenario es exactamente igual a la sección de consola, salvo que ahora tenemos en cuenta el ancho, alto e imagen de los elementos a la hora de crearlos.

También se añaden los elementos “limpiarEscenario” (que hace nula la imagen de todos los elementos del juego) y un método toString para la presentación del juego.

Clase “Elemento”

Igual que en la sección de consola, pero en este caso la imagen pasa a ser de tipo “ImageIcon” y se añaden algunos getters para ciertos atributos de relevancia.

Clase “Personaje”

De nuevo, la estructura es la misma que en la sección por consola, salvo algún pequeño detalle, como que debemos tener en cuenta que las posiciones en Swing se toman desde la esquina superior izquierda y se miden en píxeles, por lo que saltar significa una reducción en “y”, mientras que caer un aumento.

Clase “Fruta”

Exactamente igual que en consola, salvo el detalle de cambiar el tipo de la imagen a ImageIcon.

Clase “FrutaMágica”

Igual que en consola, salvo que el método desenfocar cambia la imagen del personaje en lugar de su símbolo.