Informe de Análisis y Modelamiento del Proyecto Final

YESIKA MILENA CARVAJAL DÍAZ
ANDRÉS FELIPE ZULUAGA ORTIZ

Departamento de Ingeniería Electrónica y Telecomunicaciones
Universidad de Antioquia
Medellín
05 de Abril de 2022

Índice

1. INTRODUCCIÓN

En este documento se presenta el informe completo y detallado del modelamiento para las clases que se usarán en el proyecto final del curso de Informática II. Este documento es un preámbulo para la implementación del proyecto y se pretende tener un órden esquemático de los objetos a utilizar, además de aprender bases de la programación orientada a objetos.

Se hace una descripción del juego, de los objetos que conforman este juega en sus interfaces, un análisis para tomar la desición de los objetos que requieren de clases, y un análisis para tomar la decisión de los atributos y los métodos necesarios para cada uno de los objetos de los que se decidieron que requerían clases.

2. RESÚMEN

El informe busca realizar un estudio pertinente del juego propuesto en cuanto a las tema de clases para, en base a esto, lograr hacer una ejecución eficiente del mismo. Se describe el proyecto final y los objetos que se usarán en este, se analiza a profundidad cada objeto y se toma una decisión sobre si cada uno de estos requiere una clase, con los que se toma la decisión de que requeire la clase se questiona sobre los atributos y los métodos que estos requieren.

3. OBJETIVOS

3.1. GENERAL

Realizar un análisis que nos permita hacer un modelamiento conceptual del proyecto final del curso para obtener una estructura y bases sólidas referente a las clases y objetos para una implementación eficiente del código.

3.2. ESPECÍFICOS

- Repasar los conceptos de clases, objetos, atributos, métodos e instanciamiento.
- Realizar un análisis de las clases para el proyecto final.
- Adquirir bases de la programación orientada a objetos.
- Desarrollar la capacidad analítica en la programación orientada a objetos.
- Adquirir conocimientos en la utilidad de clases en programación.

4. DESCRIPCIÓN GENERAL DEL JUEGO

El juego para nuestro proyecto final consiste en un juego tipo Bubble Shooter en el que habrá un patrón de esferas de colores aleatorios en la parte superior, y se generará una esfera de color aleatorio para lanzar desde la parte inferior (esfera a lanzar). Si la esfera de la zona a la que se lanza la esfera generada es del mismo color, se eliminarán ambas esferas al tocarse, y las esferas del mismo color que estén adheridas a la esfera de la zona tocada también se eliminarán; pero si no son del mismo color, la esfera generada se adherirá a la bola tocada.

El objetivo del juego es eliminar todas las esferas antes de que termine el tiempo. Habrán distintos niveles. La dificultad del nivel dependerá de los colores de las esferas en el patrón, y del tiempo que se tiene para completar el nivel.

La bola a lanzar se lanza presionando ésta hacia abajo generando un resorte, la cual implementará una fuerza de resorte que influirá en la velocidad y movimiento de la bola.

Los modelos físicos que se implementan en el proyecto son:

1. Movimiento parabólico.
2. Choque.
3. Resorte

5. DESARROLLO

Inicialmente para del desarrollo del proyecto se realiza prototipos de cada sección del sistema para posteriormente unir estas secciones y lograr resolver el proyecto de manera mas analítica.

5.1. Objetos:

Los objetos visibles que estarán en la interfáz del proyecto serán 4 fondos, 3 botones para indicar el nivel que se desea jugar, un botón para volver al menú de inicio, y un botón para resetear el nivel, una esfera lanzada, un arco en donde se posicionará la esfera lanzada en su estado de reposo, un resorte que dará cuenta de la variación de la posición de la esfera lanzada con respecto a su estado de reposo, dos barreras que limitarán el movimiento de la esfera lanzada, un vector de visualización de la trayectoria inicial de la esfera que se lanza, varios objetos de esfera, un temporizador, un mensaje de victoria y un mensaje de derrota.

Los objetos se visualizan en las figuras XXXXX-XXXXXX.

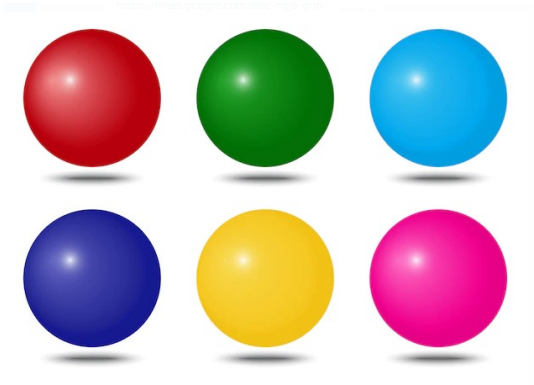

Figura 1: Objeto de bolas.

[1]

Figura 2: Objeto de barra.

[2]

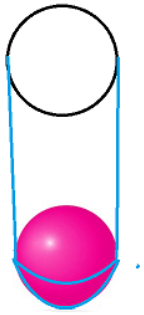

Figura 3: Obejtos de bola para lanzar, arco y resorte.

Figura 4: Objeto de botón.

Habrán 4 interfaces. La primera interfaz será la del menú de inicio y se mostrará inmediatamente ejecutar el programa y será en la que se elija el nivel por medio de los botones. Respecto al botón elegido, se mostrará alguna de las otras tres interfaces(una por cada nivel), las cuales serán las interfaces de nivel 1, 2 y 3.

Las interfaces descritas se muestran en las figuras XXXX-XXXX.

Figura 5: Interfáz del menú de inicio.

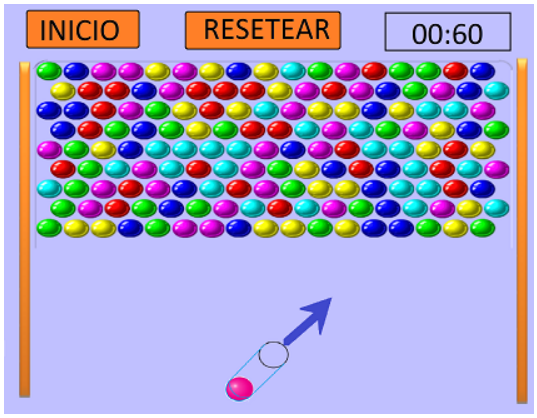

Figura 6: Ejemplo de la interfáz de nivel.

Las tres interfaces de nivel diferirán en el fondo, en la distribución de las bolas, y en que en los niveles 2 y 3 no se mostrará la flecha de direccionamiento.

5.2. Clases:

A continuación se hará un análisis de los objetos para observar si requieren de una clase.

5.2.1. Fondos:

Los fondos variarán según el botón que se elija y según en qué momento se elija. Sus acciones serán aparecer o desaparecer. Esto se puede realizar utilizando el sort de los botones y no requieren ninguna característica de la programación orientada a objetos, como ninguna característica de la física, por lo que no requieren de una clase.

5.2.2. Botones:

Los botones aparecerán o desaparecerán según la interfáz en la que se esté, o sea que también dependerán de los botones que se elija y según en qué momento se elija, por lo que tampoco requiere una clase.

5.2.3. Arco:

El arco aparecerá en las interfaces de niveles y desaparecerá en la interfáz del menú inicial. Este objeto dependerá de los botones que se elija y según en qué momento se elija, por lo que no requiere una clase.

5.2.4. Resorte:

El resorte se compondrá de 2 líneas rectas, en las cuales una irá desde extremo izquierdo del arco hasta el extremo izquierdo de la bola a lanzar, y la otra línea irá desde el extremo derecho del arco hasta el extremo derecho de la bola a lanzar. Este objeto aparecerá cuando se está en una interfáz de nivel y se presione la bola, y desaparecerá en la interfáz del menú inicial o cuando la bola no esté presionada. Además, el resorte se extenderá, extendiendo el punto final de sus líneas rectas (o sea el punto en la bola de lanzamiento). El punto inicial de las líneas rectas del resorte será constante, ya que el arco es constante; el punto final de las líneas serán el punto inicial de estas (constante) menos la diferencia entre las posiciones del arco y de la esfera de lanzamiento (constante menos variable), o sea que depende de dos constantes y una variable, la cual sería la posición de la esfera de lanzamiento. Ya que el resorte variará su posición final

dependiendo de una característica física de este, se modela como clase. Además, la extensión del resorte dará el valor de la fuerza imprimida a la esfera a lanzar.

5.2.5. Esfera:

La esfera a lanzar tendrá cuatro movimientos diferentes. Uno de ellos será al inicio del nivel, cuando se encuentra inmóvil en su posición de equilibrio en la misma posición del arco. El segundo movimiento será cuando se presione el mouse sobre este, tomando la posición del mouse hasta que la extensión del resorte llegue a su máximo, el tercer movimiento es cuando la vola está viajando por la interfaz una vez se suelte el mous y habiendo generado fuerza por medio del resorte, el cuarto movimiento será cuando toque una bola de color diferente en el cual se quedará inmóvil en la posición del choque, y el quinto es cuando toque una bola de un color igual o una bola del mismo color la toque, en cuyo caso caerá en caída libre. Además, si este objeto choca con el objeto de la barra, su posición variará respecto a este choque. Como hay movimientos físicos y amplios para este objeto, ya que su tercer movimiento es parabólico, además de que se pueden producir choques con las barras, se creará una clase para modelar las posiciones de este con respecto a sus velocidades, aceleraciones y fuerzas generadas sobre ella.

5.2.6. Barreras:

Las dos barreras aparecerán en las interfaces de niveles y desaparecerá en la interfáz del menú inicial. En las interfaces de nivel serán objetos estáticos. Este objeto dependerá de los botones que se elija y según en qué momento se elijan, por lo que no requiere una clase.

5.2.7. Vector de direccionamiento:

La vector de direccionamiento tiene un funcionamiento semejante al del resorte, sus atributos y métodos son semejantes. Este objeto variará su ángulo y su longitud, lo cual se modula con la variación en su punto final. Ya que tiene un funcionamiento parecido al del resorte, se generará otra clase para este. Este objeto se visualizará sólamente en el nivel 1.

5.2.8. Esferas:

Las esferas se visualizarán en los niveles, y permanecerán inmóviles hasta ser tocados por una bola de lanzamiento del mismo color, o tener una secuancia de bolas de igual color que toque con esta esfera de lanzamiento. Una vez suceda este caso, las bolas en la secuencia caerán en caída libre.

5.2.9. Temporizador:

El objeto temporizador se genera cuando se esté en una interfaz de nivel. Su tiempo inicial depende del nivel. Como no tiene funcionamientos físicos depen-

dientes de otras variables, no se creará una clase para este.

5.2.10. Mensajes:

Los mensaje de victoria y mensaje de derrota se mostrarán cuando se gane el nivel o cuando se pierda, respectivamente; el resto del tiempo estará invisible. Este no requiere de una clase ya que simplemente aparecerá o desaparecerá.

En conclusión, se generarán 4 clases; para los objetos de resorte, esfera a lanzar, vector de direccionamiento y esfera.

5.3. Atributos y métodos:

En esta sección se hace un análisis de los atributos y los métodos que se espera se realicen en la práctica, y se especifica el porqué se realizan estos y se descartan otros posibles métodos o atributos.

Se hace la aclaración de que a todas las clases se les creará el método constructor para inicializar las variables de sus atributos, por lo que no se especificará en las subsecciones siguientes.

5.3.1. Resorte:

Atributos:
El resorte variará la posición final de sus líneas izquierda y derecha en x y en y para así modificar su ángulo y su longitud, y esta posición dependerá de los atributos de la esfera a lanzar, por lo que sus atributos y métodos serán con respecto a su posición.

La posición final de las líneas del resorte dependen de la posición de la esfera a lanzar, la cual se calcula por medio de los métodos de esta clase, y como no hay dependencia de más variables, el atributo de la clase de vector de direccionamiento será el de su posición final en x para la línea izquierda y su posición final en y para la línea izquierda, su posición final en x para la línea derecha y su posición final en y para la línea derecha.

Métodos:
Se necesita calcular la posición final de las dos líneas que forman el resorte, por lo que se necesitan dos métodos, uno para calcular la posición de la línea izquierda y el otro para calcular la posición de la línea derecha. La posición de ambas depende de la posición de la esfera a lanzar, por lo que se necesitan métodos de Get y de Set para, respectivamente, obtener y reescribir el valor de estos atributos.

5.3.2. Esfera a lanzar:

Atributos: La esfera a lanzar variará en su color y en su posición, sin embargo su color se genera de forma aleatoria, por lo que sus atributos y métodos serán con respecto a su posición, ya que esta sí depende de otras variables.

Cuando el movimiento de la esfera a lanzar sea parabólico o de caída libre, su posición dependerá de su posición inicial (constante), de su velocidad y de su aceleración (variables); y cuando la bola esté inmóvil, esta tendrá una velocidad y una aceleración iguales a 0, con lo que su posición siempre será la inicial.

Como la posición inicial será una constante, y las velocidades y aceleraciones serán variables, habrá que generar atributos para los dos últimos, por lo que los atributos de esta serán la posición en x, la posición en y, la velocidad en x, la velocidad en y, la aceleración en x y la aceleración en y.

Métodos:
Se necesita calcular la posición de la esfera a lanzar, por lo que se necesita del método de posición para calcularlo. La posición depende de la velocidad y de la aceleración, por lo que se necesitan métodos de Get y de Set para, respectivamente, obtener y reescribir el valor de estos atributos; además se necesitan los métodos de velocidad y aceleración para calcular estos.

5.3.3. Vector de direccionamiento:

Atributos:
El vector de direccionamiento variará su posición final en x y en y para así modificar su ángulo y su longitud, y esta posición dependerá de los atributos de la esfera a lanzar, por lo que sus atributos y métodos serán con respecto a su posición.

La posición final del vector de direccionamiento dependen de la posición de la esfera a lanzar, la cual se calcula por medio de los métodos de esta clase, y como no hay dependencia de más variables, el atributo de la clase de vector de direccionamiento será el de su posición final en x y su posición final en y.

Métodos:
Se necesita calcular la posición final del vector de direccionamiento, por lo que se necesita del método de posición para calcularlo. La posición depende de la posición de la esfera a lanzar, por lo que se necesitan métodos de Get para su posición en x y en y, y de Set para su posición en x y en y para, respectivamente, obtener y reescribir el valor de estos atributos.

5.3.4. Esfera:

Atributos: Las esferas variarán en su color y en su posición, sin embargo su color se genera de forma aleatoria, por lo que sus atributos y métodos serán con respecto a su posición, ya que esta sí depende de otras variables.

Cuando el movimiento de la esfera sea de caída libre, su posición en el eje x será constante, y en el eje y su posición dependerá de su posición inicial en y, de su velocidad en y y de su aceleración en y; y cuando la bola esté inmóvil, esta tendrá una velocidad y una aceleración iguales a 0, con lo que su posición siempre será la inicial.

Como la posición inicial será una constante, los atributos de esta serán la posición en y, la velocidad en y, y la aceleración en y.

Métodos:
Se necesita calcular la posición de la esfera sólo en el eje y, por lo que se necesita del método de posición para calcularlo. La posición depende de la velocidad y de la aceleración, por lo que se necesitan métodos de Get y de Set para, respectivamente, obtener y reescribir el valor de estos atributos; además se necesitan los métodos de velocidad y aceleración para calcular estos.

6. CONCLUSIONES

- Para la implementación del juego del proyecto final se tomó la conclusión de que se usarán cuatro clases: una para la esfera a lanzar, una para las esferas, una para el vector de direccionamiento y otro para el resorte.
- Se planteó un esquema de los objetos que serán usados, haciendo un concepto de código, que luego será refinado al desarrollar el juego.
- Se establecieron todas las imágenes que serán usadas y harán parte del juego, tales como las esferas, y las paredes mismas, así como los respectivos botones de cada interfaz.
- Se analizó el comportamiento de la esfera que será lanzada, y así pudimos concluir cómo funcionará su movimiento.

7. REFERENCIAS

Referencias

[1]

[2] investigaresfacil.wordpress.com. GrÁfico de barras verticales. [Online]. Available: https://investigaresfacil.wordpress.com/2015/03/25/grafico-de-barras-verticales/