Instituto Tecnológico de Costa Rica

Ingeniería en Computación

IV Tarea Programada

Steven Peraza Porras

Jafeth Vázquez Herrera

Sede San Carlos

Noviembre 2017

Tabla Contenidos

Introducción

Este tarea programada trata sobre la creación de una adaptación del popular juego llamado “Battle City” de la consola Nintendo Entertaiment System (NES), el cual trata sobre, tal y como lo indica su nombre, de una batalla de tanques, los cuales tienen el objetivo de destruir la base del jugador, por lo cual la meta del jugador es destruir los tanques enemigos antes que la base sea destruida. Sin embargo, para la realización de este trabajo se pide que se cambie esta mecánica, es decir, que se inviertan los papeles, esta vez que sea el jugador que destruya las bases enemigas y los tanques enemigos deben de proteger sus fuertes de los ataques del jugador.

Para lograr esto, se debe implementar una aplicación web en el lenguaje de programación Javascript, mediante la utilización de librerías gráficas para su visualización correcta en los navegadores de internet (ya sea Mozilla Firefox, Google Chrome, Opera, Microsoft Edge o cualquier otro que soporte este lenguaje), con este algoritmo se podrá recorrer la matriz de juego buscando las manzanas utilizando la menor cantidad de movimientos.

Análisis del Problema

En este proyecto se solicita hacer un juego en Javascript de tanques el cual debe contar con las siguientes caracteristicas:

-El usuario debe manjar un tanque con el objetivo de destruir las bases enemigas.

-Deben existir tanques enemigos que traten de dificultarle al usuario cumplir su objetivo.

-Se solicitan 3 tipos de tanques enemigos diferentes.

-Deben generarse 3 niveles antes de completarse el juego, los cuales deben ser aleatorios.

-Los niveles deben tener ladrillos indestructibles en los bordes y ladrillos esparcidos por el mapa los cuales pueden ser destruidos por el tanque del usuario pero no por los tanques enemigos.

Además de esto es necesario trabajar con ciertas fisicas basicas para que el funcionamiento del juego sea correcto. Sobre movimiento de los objetos deben manejarse correctamente las colisiones entre objetos. Además los objetos se deben destruir, cuando una bala los deja sin vida o bien las balas mismas cuando impactan con un objetivo.

Durante el juego hay varios procesos simultaneos por lo que de alguna manera se deben manejar diferentes hilos para que el juego pueda funcionar y que un proceso no se interponga con otro.

El juego termina ya sea cuando el usuario completa los 3 niveles ganando así el juego o bien cuando este se queda sin vidas y pierde.

Solución del Problema

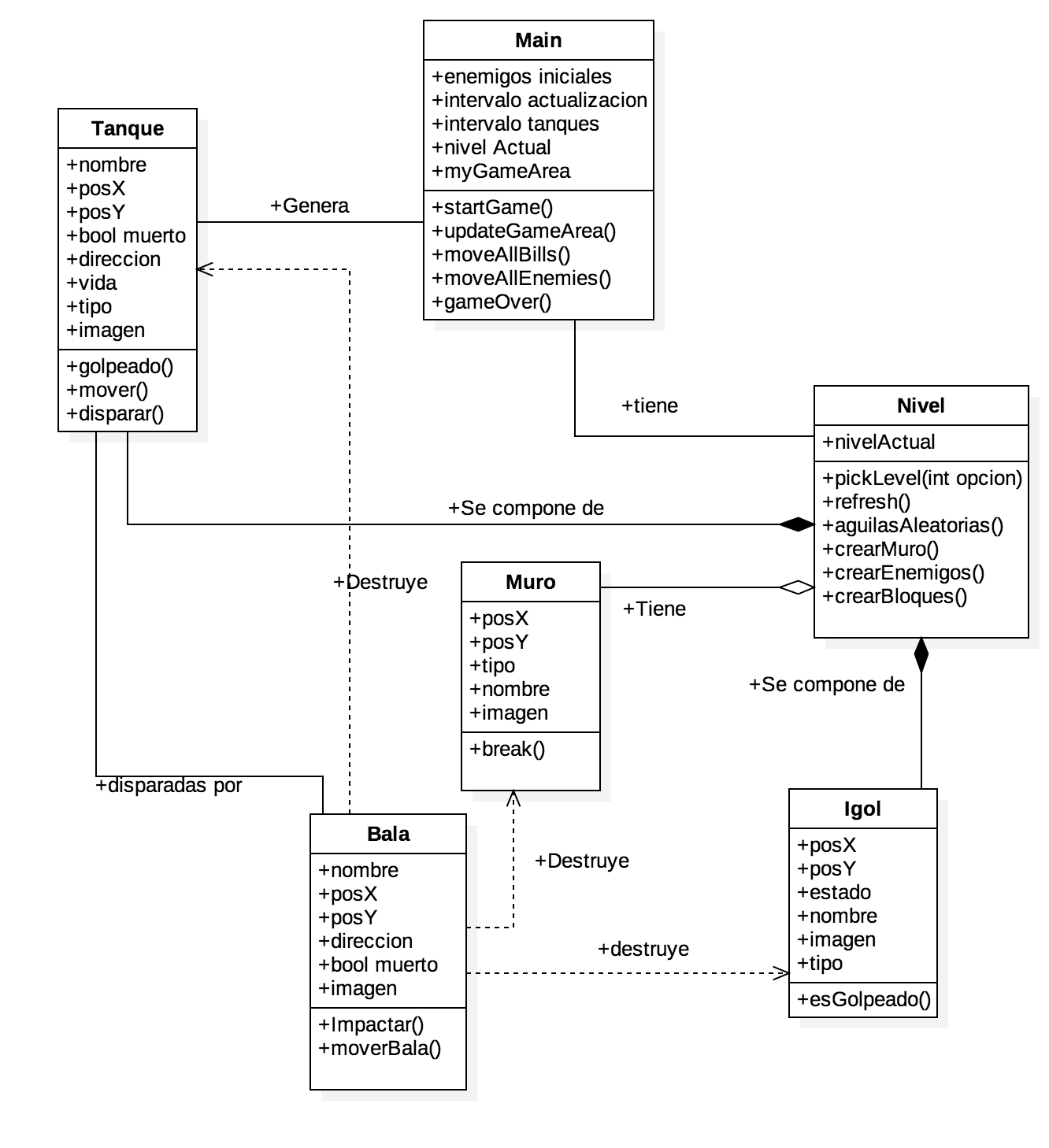


Ilustración 1.1 Diagrama de Clases UML

En la solucion implementada se utilizo Canvas para el manejo de elementos gráficos en javascript, además se utilizaron intervalos para la simulación de los hilos. Particularmente en este proyecto tenemos 3 procesos principales.

El primero es el intervalo de actualizacion el cual se encarga de verificar que cambios se deben hacer en la matriz y pintar la misma cada 60 milisegundos. Este se encarga tambien del movimiento de las balas, escuchar las teclas y mover el tanque del jugador.

El segundo es el intervalo de creación de enemigos. Este se encarga de crear nuevos enemigos en posiciones aleatorias cada 10 segundos.

El tercero es el intervalo de movimiento de los tanques enemigos. Este se encarga de mover aleatoriamente los tanques enemigos cada segundo y de disparar si se cruzan de alguna manera con el tanque del jugador.

Con respecto a los elementos existen los tanques de los cuales hay 3 tipos, posteriormente hablaremos de ellos con más detalle, objetivos (igols) los cuales son los que hay que destruir para ganar un nivel, de estos hay 2 tipos y poseen unicamente 1 de vida. Las balas son proyectiles disparados por los tanques y poseen una direccion en la cual se desplazan, estas se destruyen al impactar con algun objeto, le reduce en 1 la vida a los tanques y destruye los bloques de ladrillo y los objetivos, no destruye bloques de metal ni el agua (bordes).

Como se puede observar en la figura 1.1 en la página anterior para la implementación del proyecto se usaron 6 clases las cuales describiremos a continuacion:

* **Main:** Esta es la clase principal y la que se encarga de inicializar los procesos. Crear los niveles y el area de juego. Y manejar el movimiento de los elementos. Esta es la clase principal.

A continución una pequeña descripción de los métodos del main:

1. **startGame():** Este método se encarga de inicializar el juego, define los intervalos de actualizacion de la pantalla, inicializa la matriz de objetos de myGameArea, crea los muros y tanques aleatorios.
2. **updateGameArea():** Este método recorre la matriz y dibuja en la pantalla los objetos según la matriz myGameArea.
3. **moveAllBills():** Este método es el que se encarga de recorrer la lista de balas activas y moverlas.
4. **moveAllEnemies:** Este método es el que se encarga de recorrer la lista de tanques aún vivos y moverlos en direcciones aleatorias. Además si se topa en la misma columna o fila que el tanque del jugador hace que el enemigo apunte hacia el y dispare.
5. **gameOver():** Se encarga de finalizar el juego cuando el jugador haya eliminado todos los objetivos y verifica si es el ultimo nivel o no. Si no lo es, pasa al siguiente nivel o bien si es el ultimo nivel le muestra al usuario que ha ganado el juego.

* **Tanque:** Los tanques son los elementos más interactivos del juego, estos se mueven por el mapa y disparan. Tanto los enemigos como el jugador son tanques, pero de tipos diferentes. Solo las balas que provienen del jugador pueden destruir un objetivo.

Existen 3 tipos de tanque y la principal diferencia entre estos son su vida. Los tanques más fuertes tienen 4 de vida, los intermedios tienen 3 de vida y los más debiles 2.

A continuacion una pequeña descripcion de los métodos de Tanque:

1. **Golpeado/Hit**Este método determina la reaccion del tanque cuando este es golpeado por una bala, realiza las comprobaciones necesarias y además le resta la vida correspondiente.
2. **Mover(direccion):** Este método recibe como parametro la direcion en la cual moverse y realiza las comprobaciones y operaciones para realizar el movimiento en la matriz de objetos.
3. **Fire():** Metodo que realiza la accion de disparar. Instancia una bala con la direccion en la que se encuentre el tanque y con el tipo de este.

* **Bala:** Las balas son proyectiles lanzados por los tanques y son los únicos objetos capaces de destruir a otros. Las balas son disparadas y por tanto instanciadas por los tanques. Cuando estas colisionan con un objeto lo destruirá inmediatamente o le restará un punto de vida.

A continuacion una pequeña descripcion de los metodos de la clase bala:

1. **Impactar():** Este metodo es el que se encarga de determinar la reaccion de la bala cuando colisiona con un objeto. Verifica si está impactando con bloque, con un tanque o con un objetivo y ejecuta la accion correspondientes desde la perspectiva de la bala.
2. **moverB():** Este metodo hace las validaciones correspondientes y determina el comportamiento de la bala miesntras de mueve..

* **Nivel:** Los niveles son las estructuras fisiscas en las que se desarrolla el juego. En el se encuentran los bloques y los tanques. Aquí es donde se encuentran los metodos que posicionan de manera aleatoria los elemento al iniciar el juego.

Los metodos de esta clase son:

1. **pickLevel():** Recibe como como parametro una un entero con el numero de semilla para el nivel actual.
2. **refresh():** Este es el método que se encarga de dibujar en la pantalla los objetos representados en la matriz logica.
3. **objetivosAleatorios()/aguilasRan:** Genera objetivos en posiciones aleatorias.
4. **crearMuro()/trumpApproves:** Este método se encarga de construir el muro de los bordes (agua).
5. **creaEnemigos():** Este método crea los tanques enemigos en posiciones aleatorias.
6. **creaBloques():** Metodo que crea los bloques de tipo aleatorio en posiciones aleatorias.

* **objetivos/Igol:** Son los objetivos que el usuario debe destruir para ganar el juego, es una clase simple que posee unicamente una posicion X y Y y una imagen de sprite. Su metodo esGolpeado()/conectado cambia el estado del objetivo y su imagen pasa a representar que ha sido destruido.
* **Muro/bloque:** Esta es una clase simple que posee posicion X y Y y un tipo y su unico metodo es break el cual destruye el bloque.

Análisis de Resultados

|  |  |  |
| --- | --- | --- |
| Descripción | Estado | Observaciones |
| Generacion de la matriz y los niveles aleatorios | 100 % completado |  |
| Manejo de Procesos simultaneamente | 100 % completado |  |
| Manejo de colisiones entre objetos | 100 % completado |  |
| Interaccion entre las balas y objetos | 100 % completado |  |
| Finaliza correctamente el nivel | 100 % completado |  |
| Finaliza el juego | 100 % completado |  |

Conclusiones

Mediante la realización de este proyecto, se pudo comprobar las ventajas y beneficios de ciertas características de la programación orientada a objetos, como por ejemplo, por medio de la herencia, se puede eliminar código redundante y extender el uso de clases existentes, por medio de la modularidad se pueden construir programas a partir de módulos de trabajo estándar que se comunican entre sí, en lugar de tener que empezar desde cero la escritura del código, lo que implica un ahorro en el tiempo de desarrollo y a una productividad mayor. Además, es posible tener múltiples copias de un objeto que coexisten sin ninguna interferencia, cada uno independiente del otro; por lo cual, la carga del trabajo puede ser fácilmente dividida en distintas clases y objetos para que la comunicación entre objetos, la cual es muy sencilla y adaptativa en este paradigma, permita que los programadores logren sus objetivos con mayor libertad y facilidad.

De hecho, en la actualidad, el paradigma de orientación a objetos es sin lugar a dudas el más utilizado por las empresas de todo el mundo a la hora de encarar desarrollos de aplicaciones de software, ya que permite representar de manera relativamente simple modelos y realidades muy complejas y esto hace que el software sea más fácil de programar, comprender y mantener.

Luego de haber realizado este proyecto, el nivel de conocimiento adquirido en el tema de objetos, IA, interfaz gráfica y otros temas relacionados se vieron notoriamente incrementados. Además se aprendieron elementos, funciones y operadores de la programación web mediante la utilización del lenguaje Javascript, uno de los 3 pilares fundamentales de este tipo de programación, gracias a lo cual se pudieron conocer sus características y ventajas, las cuales en la actualidad empiezan a ganar más terreno en las empresas y el ambiente laboral en general.

Para terminar, se experimentaron y recordaron diferentes técnicas del paradigma de la programación orientada a objetos, la cual es entretenida e interesante de utilizar, además, de que estos nuevos y viejos conocimientos en este apartado pueden que nos sirvan más adelante en nuestra carrera universitaria y en un futuro ambiente laboral.

Bibliografía