Instituto Tecnológico de Costa Rica

Escuela de Computación

Lenguajes de Programación

Profesor

Oscar Víquez Acuña

Cuarto Proyecto

(Paradigma Orientado a Objetos)

**“Battle City TEC”**

Josué Arce González

Eliomar Rodríguez Arguedas

Sede San Carlos

12 de noviembre del 2017

# **Tabla de contenidos**

# 

[**Tabla de contenidos**](#_rmx61jd94qc4) **1**

[**Introducción**](#_nts2g4t3cunq) **2**

[**Análisis de problema**](#_di9t5oo99zb2) **3**

[**Solución del problema**](#_lzjkc29f5pso) **4**

[**Análisis de resultados**](#_omk6cmutg5qj) **5**

[**Conclusiones**](#_10azkripcqxe) **6**

[**Recomendaciones**](#_if02ub1mgqiv) **7**

[**Bibliografía**](#_schowjxtuocs) **8**

# 

# 

# **Introducción**

**Battle City** es un videojuego virtual creado por primera vez en 1985 por Namco, una empresa japonesa orientada al entretenimiento público, este juego fue destinado para las consolas Nintendo, GameBoy, entre otras, fue un gran invento por parte de los mencionados, ya que millones de personas alrededor del planeta utilizaban alguna de estas consolas para jugar dicho juego.

En el presente documento se muestra el estado actual del videojuego creado, como funciona, lógica utilizada, y medidas que se tomaron para lograr el objetivo, además, se cuenta con un apartado donde se mencionan recomendaciones personales para futuros programadores o personas que desean diseñar o crear un videojuego similar a este en la plataforma **WEB**.

Cabe destacar que el lenguaje de programación utilizado para realizar este proyecto fue JavaScript, este lenguaje está completamente diseñado para ser utilizado en el sector WEB, fue ingeniado por Brendan Eich, en los años 90 para poder así tener una mejor condición de vida en relación a la navegación web, ya que en esos tiempos el acceso y el retorno de páginas era complejo y además pesadas, por lo que pensó en una solución para terminar con este tipo de problemas, por lo que creó **JavaScript.**

# **Análisis de problema**

# El juego a desarrollar es similar al mencionado juego,pero con algunas variantes en donde existen varios tipos de tanques(Héroe y Enemigos), el que es manejado por el usuario y otros que cuidan los objetivos que el tanque del usuario debe destruir. Siendo así y con el propósito de explotar al máximo las fortalezas del paradigma, las siguientes serán las características básicas del juego:

# 1. El juego posee 3 niveles de juego, por cada nivel las piezas **BLOQUE, OBJETIVOS, TANQUES ENEMIGOS** deben ser posicionados en el mapa de manera aleatoria, el juego acaba cuando todos los **OBJETIVOS** hayan sido destruidos.

# 2. En el mapa deberán aparecer 3 tipos de **TANQUES ENEMIGOS** los cuales deberán intentar defender los objetivos del mapa, evitando que el tanque héroe los destruya, en el juego deben existir 3 tipos de tanque, los cuales poseen atributos o características distintas y notables, conforme corre el tiempo deben ir apareciendo más y más tanques en el mapa.

# 3. Hay 2 tipos de bloques:

* **Bloques Normales:** pueden ser destruidos por el tanque héroe pero no por los tanques enemigos.
* **Bloques Borde:** ningún tanque puede destruir este tipo de bloque, ya que hacen referencia al borde del mapa, y su función es evitar que se salgan del mapa.

# 4. El tanque héroe posee 3 vidas, las cuales pueden llegar a 0 si un tanque enemigo lo destruye, además, los tanques pueden moverse en 4 direcciones (Arriba, Abajo, Izquierda,Derecha), cuando un tanque dispara una **BALA,** la misma solo puede desplazarse en la dirección a la que el tanque se encontraba mirando.

# 5. Todos los objetos deben tener métodos relacionados con la interacción con otros objetos. No olvidarse que las balas también deben ser objetos.

# 6. Deben considerar la existencia de objetos que controlen el juego y las interacciones entre otros objetos.

# 

# 

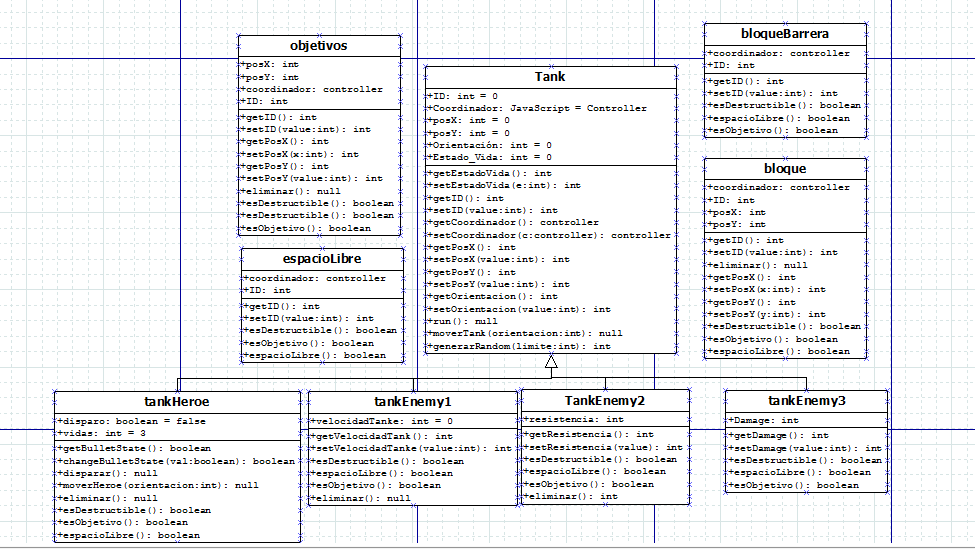
# 

# **Solución del problema**

A continuación se menciona la lógica utilizada para alcanzar el objetivo y sus pasos necesarios:

1. Declaración de las clases: cada clase juega un papel muy importante, ya que cada una hace referencia a un objeto en el mapa, la cual además posee atributos necesarios para que el objeto pueda cumplir con su labor; por ejemplo: **bala:** esta clase hace referencia a la bala que un tanque **X** dispara en un momento determinado.
2. Creación de la matriz lógica: este apartado es el más importante de todos, ya que es la base del programa, el juego se construye dentro del mismo y todo ocurre en el susodicho, cualquier cambio realizado debe ser reflejado primero en esta matriz, por lo que adquiere gran importancia.
3. Movimiento de los objetos: para poder realizar el efecto de que un tanque se mueve tanto solo como por el mismo usuario se debió crear una función que tradujera la matriz lógica en dibujos o figuras notables a la vista, para sí generar un mejor ambiente donde el usuario se sintiese que de verdad está jugando, cuando un elemento de la matriz lógica se cambia de posición este se denota de manera inmediata en pantalla.
4. Tiempo: el juego tiene un máximo de 2:00 minutos para jugar, cuando el temporizador llegue a 0 el juego termina y el usuario habrá perdido cuando eso, cada vez que el juego se reinicia el temporizador vuelve a 2:00 minutos, cabe mencionar también que cada 15 segundos aparecerán más y más tanques enemigos, por lo que al final del juego si llegase a 0 el temporizador habrán 8 tanques nuevos, 11 en total.
5. Objetivos: es un objeto más en el mapa, solamente que es el principal objetivo del usuario, deben ser destruidos para poder avanzar de nivel, una vez alcanzado el nivel 3 y al ser destruidos los objetivos el juego termina, el jugador habrá **ganado.**

Se presenta el diagrama de clases del programa:



A continuación se detallan los métodos principales utilizados en la parte lógica:

1. crearMatriz(): permite generar la matriz lógica junto con todos sus elementos que conforman el juego, tales como: tanques, objetivos,balas, bloques.
2. generarPosicionAleatoria(): permite generar un número entre 1 y 14(tamaño de la matriz).
3. restarObjetivos(): permite eliminar un objetivo luego de haberlo eliminado.
4. reiniciar Juego(): reinicia todas las variables y listas del juego luego de haber pasado de nivel.
5. terminarJuego(estado): se llama cuando el usuario ha perdido o ha ganado.
6. reprMusica(opción): Función encargada de la reproducción de canciones o audios del juego.
7. verificarEstadoJuego(): Permite saber en qué nivel se encuentra el usuario, cada vez que avanza reinicia el juego.
8. setObject(X,Y,Objeto): pone un nuevo objeto en una posición exacta.
9. getObject(X,Y): obtiene el objeto que se encuentra en una posición exacta.
10. disparar(X,Y,Pertenece,Orientación): se acciona cuando el usuario dispara una bala, este avanza en una columna/fila de la matriz.
11. BorrarEnemigo(tanke,estado): elimina el objeto enemigo del mapa.
12. getHeroe():obtiene el heroe del juego.
13. actualizar(): pinta la matriz lógica con figuras e imágenes en pantalla para que el usuario pueda apreciar el juego.
14. addNewEnemy(): crea un nuevo enemigo en una posición [X,Y] y lo agrega a la lista.
15. startTimer(duracion,objeto): inicializa el temporizador.
16. quitarBalasMatriz(tanke): remueve el tanque de la matriz lógica.
17. timer(): permite crear el temporizador.

# **Análisis de resultados**

|  |  |  |
| --- | --- | --- |
| Requisito | Estado | Observaciones |
| Creación de la matriz | Completo | N/A |
| Movimiento del tanque héroe | Completo | N/A |
| Disparar con el tanque héroe | Completo | N/A |
| Movimiento de los tanques enemigos | Completo | N/A |
| Disparo por parte de los enemigos | Completo | N/A |
| Creación de los objetivos, bloques, borde | Completo | N/A |
| Uso del temporizador | Completo | N/A |
| Gestión del juego en general | Completo | N/A |
| Resultado general | Completo | N/A |

# **Conclusiones**

El proyecto se encuetnra en un estado muy completo, todas las funciones solicitadas cumplen y se encuentran en funcionamiento; es satisfactorio ver como todo esfuerzo tiene su remuneración, el proceso para iniciar con el proyecto fue algo complejo, no se sabía con exactitud cómo funcionaba un hilo en web, específicamente en JavaScript, esto debido a la falta de conocimiento en este lenguaje, después de varias horas de búsqueda, concluimos que el **setInterval** no es en sí un hilo pero realiza llamadas funciones elaboradas cada cierto tiempo, por lo que decidimos optar por su uso, este funciona de maravilla y cumple con lo requerido.

Otro punto que vale recalcar es el hecho del uso del **MVC,** no se explotó este patrón de diseño como se hubiese gustado pero el programa tiene cierta forma y se basa en este para poder trabajar, JavaScript al ser un lenguaje amplio y sin fronteras le permite al programador realizar lo que desee, inclusive al ser un lenguaje no tipado permite la manipulación de variables de una forma muy diferente a la comúnmente vista en otros lenguajes tales como C# o JAVA.

# **Recomendaciones**

Se enumeran ciertas recomendaciones escritas para futuros programadores que vayan adentrarse tanto en el lenguaje JavaScript como en la elaboración del juego en sí:

1. Utilizar el patrón de diseño **MVC,** esta forma de programación es muy útil y le permite al usuario tener un marco de trabajo mucho más estable y ágil, apartando la vista de la parte lógica, y teniendo un controlador el cual es el intermediario entre los mencionados.
2. Reutilizar funciones: si se crea una función y esta funciona para muchos casos debe ser usada en lugar de crear otra o realizar otra forma, siempre es bueno tener poco código.
3. Hay mucha documentación: si eres un novato en este lenguaje como nosotros, no te preocupes que existen muchas formas de aprender esta rama, tanto en internet como en libros, JavaScript es un lenguaje popular actualmente, por lo que se le está dando mucho enfoque y las empresas están explotando a las personas que tengan conocimientos en esta.
4. Trabajar con tiempo: un error garrafal es trabajar bajo presión, por lo que es mejor ir trabajando con tiempo para ir obteniendo resultados, en vez de estar apresurado días antes de la entrega del proyecto, siempre es bueno tener un plan de trabajo.

# 

# **Bibliografía**