Proyecto final

Juan Esteban Grajales Carmona
Brayan Steven Avila Marin

Despartamento de Ingeniería Electrónica y
Telecomunicaciones
Universidad de Antioquia
Medellín
Abril de 2022

Índice

1. Introducción

Eres un piloto espacial que defiende la tierra de los invasores alienígenas, cuando un terrible enemigo aparece y destruye tu nave tendrás que repararla y emprender un viaje para detener al enemigo, enfréntate a los peligros del espacio, persigue al terrible alienígena y defiende a la humanidad, a través de tres niveles pon a prueba tus habilidades para avanzar esquivando la basura espacial, peleando contra los enemigos y detener al alienígena que pone en riesgo al planeta tierra.

2. Objetivo general

Se tiene como objetivo llegar al punto en el que las clases y el propio juego pueda correr de manera autónoma es decir sin errores de por medio en el mismo, se tiene claro por parte de los programadores qué las clases deben estar bien definidas con sus parámetros privados y públicos en relación al mismo juego además de tener conocimiento asimilado acerca de las prácticas de laboratorio realizadas en clase es por ello qué se le ha dado la estructura anteriormente mencionada al tema de las clases pues se consideran las necesarias para el desarrollo de dicho proyecto esto teniendo en cuenta que no se simplifica a una sola clase ni tampoco se excede en la creación de ellas, además de darle una óptima función a cada sin redundar en código de otra, Por consiguiente no sea de definir el mismo valor en diferentes clases, esto con el fin de tener un mejor uso de la memoria del pc y a su vez no desperdiciar recursos propios de Qt y la máquina que lo esté utilizando.

3. Conceptos clave

- Qt: es un framework multiplataforma orientado a objetos ampliamente usado para desarrollar programas (software) que utilicen interfaz gráfica de usuario, así como también diferentes tipos de herramientas para la línea de comandos y consolas para servidores que no necesitan una interfaz gráfica de usuario. Qt es desarrollada como un software libre y de código abierto a través de Qt Project, donde participa tanto la comunidad, como desarrolladores de Nokia, Digia y otras empresas. Anteriormente, era desarrollado por la división de software de Qt de Nokia, que entró en vigor después de la adquisición por parte de Nokia de la empresa noruega Trolltech, el productor original de Qt, el 17 de junio de 2008. Qt es distribuida bajo los términos de GNU Lesser General Public License y otras. Por otro lado, Digia está a cargo de las licencias comerciales de Qt desde marzo de 2011. 1

Figura 1: Logo de Qt

- interfaz gráfica: La interfaz gráfica de usuario, conocida también como GUI (del inglés graphical user interface), es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo de una máquina o computador. Habitualmente las acciones se realizan mediante manipulación directa, para facilitar la interacción del usuario con la computadora. Surge como evolución de las interfaces de línea de comandos que se usaban para operar los primeros sistemas operativos y es pieza fundamental en un entorno gráfico. Como ejemplos de interfaz gráfica de usuario, cabe citar los entornos de escritorio Windows, el X-Window de GNU/Linux o el de Mac OS X, Aqua. En el contexto del proceso de interacción persona-computadora, la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático. 2

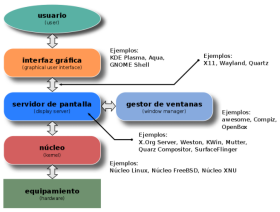

Figura 2: Diagrama de una interfaz grafica

- clase: En informática, una clase es una plantilla para la creación de objetos de datos según un modelo predefinido. Las clases se utilizan para representar entidades o conceptos, como los sustantivos en el lenguaje. Cada clase es un modelo que define un conjunto de variables y métodos apropiados para operar con dichos datos. Cada objeto creado a partir de la clase se denomina instancia de la clase. La programación orientada a objetos es la base principal para los tipos de objetos. Permiten abstraer los datos y sus operaciones asociadas al modo de una caja negra. Los lenguajes de programación que soportan clases difieren sutilmente en su

soporte para diversas características relacionadas con clases. La mayoría soportan diversas formas de herencia. Muchos lenguajes también soportan características para proporcionar encapsulación, como especificadores de acceso. Una clase también puede tener una representación (meta objeto) en tiempo de ejecución, que proporciona apoyo en tiempo de ejecución para la manipulación de los metadatos relacionados con la clase. 3

- objeto: En el paradigma de programación orientada a objetos (POO, o bien OOP en inglés), un objeto es un ente orientado a objetos (programa de computadoras) que consta de un estado y de un comportamiento, que a su vez constan respectivamente de datos almacenados y de tareas realizables durante el tiempo de ejecución. Un objeto puede ser creado instanciando una clase, como ocurre en la programación orientada a objetos, o mediante escritura directa de código y la replicación de otros objetos, como ocurre en la programación basada en prototipos. Estos objetos interactúan unos con otros, en contraposición a la visión tradicional en la cual un programa es una colección de subrutinas (funciones o procedimientos), o simplemente una lista de instrucciones para el computador. Cada objeto es capaz de recibir mensajes, procesar datos y enviar mensajes a otros objetos de manera similar a un servicio. En el mundo de la programación orientada a objetos (POO), un objeto es el resultado de la instancia de una clase. 4

- Programación orientada a objetos: La Programación Orientada a Objetos (POO) es un paradigma de programación, es decir, un modelo o un estilo de programación que nos da unas guías sobre cómo trabajar con él. Se basa en el concepto de clases y objetos. Este tipo de programación se utiliza para estructurar un programa de software en piezas simples y reutilizables de planos de código (clases) para crear instancias individuales de objetos. Lo que buscamos es dejar de centrarnos en la lógica pura de los programas, para empezar a pensar en objetos, lo que constituye la base de este paradigma. Esto nos ayuda muchísimo en sistemas grandes, ya que en vez de pensar en funciones, pensamos en las relaciones o interacciones de los diferentes componentes del sistema. 5

- Lore: El Lore o trasfondo de un juego es el conjunto de historias, datos, personajes, representaciones, etc que conforman el universo representado en el mismo y le dan coherencia.

- Sprite: En el desarrollo de videojuegos se utiliza sprite para denominar el mapa de bits que creaba en la pantalla un hardware gráfico especializado pero sin necesidad de usar la CPU del ordenador. Por eso, los sprites se usan para los personajes protagonistas de los videojuegos, ya que con un sprite es más fácil gestionar un personaje independientemente del fondo, es decir consiste en la máscara o la imagen que el desarrollador de videojuegos asigna a un objeto que puede colisionar. 6

Figura 3: Ejempli de sprite

- pixelart: El “arte de píxel”, más comúnmente llamado Pixel Art, consiste básicamente en una disciplina artística que utiliza una computadora y diversos programas gráficos de edición de imágenes rasterizadas para la creación de obras de arte digital, las cuales son realizadas pixel a pixel. El Pixel Art proviene de las antiguas videoconsolas, primitivos juegos para celulares y para computadoras, sin embargo en aquellos tiempos no eran consideradas obras de arte, eran las únicas formas de crear imágenes para este tipo de implementaciones disponible para los desarrolladores y el hardware de aquella época. Se empezó a considerar al Pixel Art como arte muchos años después, luego que este tipo de imágenes se dejarán de usar en el diseño de software, gracias a que el hardware como por ejemplo las tarjetas de video se volvió más poderoso, y por lo tanto capaz de procesar más cantidad de información en menos tiempo, lo que incluía a las imágenes. 7

Figura 4: Ejempli de pixelart

4. Clases

Este proyecto constará de nueve clases las cuales se retoman a continuación en el planteamiento del videojuego, estas clases serán:

- menú inicial
- mainwindow
- intro
- personaje
- basura
- disparo
- laser
- miniboss
- Boss

A continuación se dará paso a la explicación de lo que se desea hacer en cada clase a partir de los conocimientos vistos durante el curso y el análisis realizado por parte de los programadores ante este desafío:

4.1. Menú inicial

En la clase menú inicial será un formulario el cual contará con los botones para las diferentes opciones del menú inicial como jugar en solitario o multijugador, cargar la partida guardada, instrucciones, además en el formulario se le solicitará al usuario un inicio de sesión, con su respectiva contraseña para poder iniciar el juego.

4.2. Mainwindow

En el mainwindow modelamos las dimensiones de cada nivel, los cambios de escena y agregaremos los objetos que necesitemos de cada clase, así como definir los movimientos de objetos y el daño que causara las colisiones entre los enemigos y obstáculos en pantalla, se busca así tener claro los momentos donde se verán afectados los personajes producto de las mismas dificultades expuestas en el juego y que estos valores se vean reflejados propiamente en otras clases.

4.3. Intro

La clase de intro será para mostrar en pantalla al jugador la introducción a la historia del juego, es decir, darle una contextualización al usuario acerca del Lore o momento en el cual se encuentra dentro del desarrollo del juego, esta pantalla se podría asociar con las pantallas de carga o cinemáticas propias de los videojuegos actuales, en la cual buscamos que el jugador no este solo jugando si no que entienda hacia dónde va dirigida la historia .

4.4. Personaje

Esta clase contempla todos los atributos relacionados con el personaje principal es decir, el controlado por el jugador, aquí podemos evidenciar, el daño que ocasiona hacia los enemigos y objetos, además de poder tener su vida,el como esta irá cambiando a lo largo del juego empezando con tres contenedores de corazón y a medida de que reciba daño estos se irán reduciendo hasta el punto de perder completamente, así teniendo un game over instantáneo, a su vez se define su movimiento el cual será rectilíneo, su sprite, sus dimensiones dentro de la misma ventana de juego incluyendo su posición de inicio en X e Y respectivamente.

4.5. Basura

Continuando con la clase denominada basura, es aquí donde podemos evidenciar el comportamiento de los asteroides que se encuentran en la pantalla de juego, dichos asteroides tiene el objetivo de estorbar o incluso impactar al jugador, haciéndolos un elemento pasivo dentro del juego los cuales si bien no son los enemigos del juego si no se tiene cuidado con ellos pueden reducir tu vida e incluso hacerte perder en cuyo caso, se decide incluir esta clase debido a que se tiene como objetivo que el juego cuente con elementos del mundo real y así hacerlo un poco más realista dentro del concepto y a su vez que se vea un aumento en la dificultad inicial brindada solo por los enemigos, cabe resaltar la definición de sus parámetros de posicionamiento X,Y además de sus dimensiones,sprite y vida ante los proyectiles del jugador.

4.6. Disparo

Esta clase, abarca el disparo con el que cuenta el personaje principal, este mismo con su sprite,movimiento vertical y el cómo impactará ante los enemigos o asteroides que se encuentre el proyectil, cabe resaltar que este proyectil tiene un movimiento independiente del jugador, es decir que si el jugador dispara y luego se desplaza hacia izquierda o derecha el proyectil no ha de verse afectado pues se plantea como una entidad independiente.

4.7. Laser

Laser es una clase en donde definimos los parámetros de un ataque determinado con el mismo nombre allí, contamos con un código muy parecido al de la clase disparos pues ambos al ser los métodos de ataque del juego comparten ciertas similitudes entre sí, allí se le asigna su sprite, y su movimiento dentro del juego por medio de fórmulas físicas y el cómo ha de interactuar entre el entorno del juego, su velocidad de movimiento y su desplazamiento de manera exhaustiva.

4.8. Miniboss

Dicha clase contempla los aspectos en relación al mini boss o minijefe que se presenta en el juego, es visto como un punto intermedio del juego el cual a pesar de no ser el jefe final tiene una dificultad más elevada que los enemigos comunes, es por esto que en esta clase se definen sus comportamientos y mecánicas, su posición inicial en pantalla con respecto a X;Y propiamente su sprite el cual se mostrará en el juego y más importante aún su vida la cual es considerablemente más elevada que un enemigo común, el objetivo es que este jefe haga puesta en escena en un momento adelantado del juego para dar a entender al jugador su progreso en el mismo.

4.9. Boss

Es una clase muy importante pues representa al enemigo final del juego, el culmen de la aventura que se plantea, este Boss tiene la finalidad de presentar un gran reto para el jugador, pues contando con vida elevada y gran cantidad de ataques, se pone como pilar a vencer antes de concluir el juego, en esta clase se definen su sprite o imagen, su movimiento y posición donde hará aparición,además de la vida con que cuenta la cual es la más alta de todos los enemigos.

Dentro de las clases mencionadas para este proyecto anteriormente se buscará implementar lo hizo dentro del curso informática 2 del presente semestre esto con la finalidad de poder hacer funcionar un proyecto enfocado hacia un videojuego, así, haciendo que estás clases funcionen armónicamente pues dentro de ellas se definirán parámetros y ecuaciones para su comportamiento con respecto a las otras, es decir, las colisiones entre objetos, movimientos físicos propios de cada clase si lo requiere, ataque y vida del personaje así como de los enemigos, se buscará además la implementación de la interfaz gráfica de Qt en vista de una animación acorde al alcance de la misma herramienta limitada por sus parámetros, teniendo esto claro se optará por un diseño pixel art reflejado en imágenes de extensión .PNG, retratando un estilo retro de los juegos propios de la NES y el ATARI, en los cuales deseamos basar este proyecto buscando un vistazo hacia el pasado de los videojuegos.

Referencias

[1] wikipedia. Qt (biblioteca). [Online]. Available: https://es.wikipedia.org/wiki/Qt_(*biblioteca*)

[2] ——. Interfaz gráfica de usuario. [Online]. Available: https://es.wikipedia.org/wiki/Interfaz_*gráfica*_*de*_*usuario*

[3] ——. Clase (informática). [Online]. Available: https://es.wikipedia.org/wiki/Clase_(*informática*)

[4] ——. Objeto (programación). [Online]. Available: https://es.wikipedia.org/wiki/Objeto_(*programación*)

[5] M. M. Canelo. ¿qué es la programación orientada a objetos? [Online]. Available: https://profile.es/blog/que-es-la-programacion-orientada-a-objetos/

[6] R. Varela. Sprites. [Online]. Available: https://www.geekno.com/glosario/sprites

[7] tecnologia+informatica. ¿que es pixel art? tipos, técnicas. [Online]. Available: https://www.tecnologia-informatica.com/pixel-art/