





Texto

Descripción generada automáticamente

# ÍNDICE

[**ÍNDICE 2**](#_klrm2ugoukh7)

[**OBJETIVOS 3**](#_qmo8e4c9jr30)

[**INTRODUCCIÓN 4**](#_txfwu4l8ji0g)

[Programación orientada a objetos como enfoque de desarrollo 4](#_ho0k2jxt1tnb)

[El esqueleto de "Galaga" como punto de partida 5](#_ejle2duvflah)

[Elementos clave del juego 6](#_obigmk8s8hj4)

[**DESARROLLO 7**](#_f9jawnlq49jh)

[Problema de pantalla principal 7](#_xoss8cihxowm)

[Problema de que cada nivel tenga sus propias imágenes y velocidad. 7](#_q37xsgqti74j)

[Adaptar el disparo de "galaga" a "cuphead" 8](#_9z5bgl3bpsd6)

[El problema con los disparos y movimiento de los jefes 9](#_xd9a4fnj0jlq)

[Colisión de disparos 10](#_8a5k0sxoc1fs)

[Cambio de música y animación al morir. 11](#_227gh2dlr7d)

[Problemas de botón de los niveles para regresar a pantalla principal 12](#_ruurbd6add3c)

[UML 14](#_brarsr705ftw)

[**CONCLUSIONES 16**](#_4e9k2homvquh)

[**BIBLIOGRAFÍA 18**](#_339y4hotjztb)

# OBJETIVOS

El objetivo de este proyecto es desarrollar un juego de disparos hacia jefes finales con una lógica similar a los juegos de Atari, utilizando la programación orientada a objetos y basándonos en el esqueleto de otro juego llamado "Galaga". A partir de esta base, nuestro objetivo es crear un juego lo más similar posible a Cuphead.

El juego estará compuesto por tres niveles, cada uno con un jefe final al que el jugador deberá enfrentarse. El jugador contará con tan solo tres vidas para completar el juego, lo que añadirá un elemento de desafío y tensión. Al igual que en Cuphead, los jefes finales tendrán patrones de ataque aleatorios, lo que mantendrá al jugador alerta y requerirá habilidad para evadir los disparos enemigos.

Uno de los elementos clave del proyecto será la implementación de la programación orientada a objetos. Esta metodología nos permitirá organizar el código de manera estructurada y modular. Cada elemento del juego, como el jugador, los jefes finales, los disparos y los elementos del escenario, serán representados como objetos con sus propias características y comportamientos.

Además, aprovecharemos el esqueleto del juego "Galaga" como punto de partida para nuestro desarrollo. Esto nos permitirá ahorrar tiempo en la implementación de funcionalidades básicas, como el movimiento de los personajes. Sin embargo, a medida que avancemos en el proyecto, personalizaremos y modificaremos el código para adaptarlo a nuestras necesidades y lograr el estilo y la jugabilidad similares a los de Cuphead.

Un requisito importante del juego será que el jugador no pueda salir del mapa. Esto implicará establecer límites en los movimientos del personaje principal y en la generación de disparos enemigos. Asimismo, el diseño del escenario se ajustará para garantizar que el jugador se mantenga en el área de juego en todo momento.

En resumen, el objetivo de este proyecto de investigación es desarrollar un juego de disparos hacia jefes finales, inspirado en Cuphead y con una lógica tipo Atari. Utilizaremos la programación orientada a objetos y aprovecharemos el esqueleto del juego "Galaga" como base. Nuestro objetivo final es lograr un juego desafiante y entretenido, con tres niveles, tres vidas, disparos aleatorios por parte de los jefes y la restricción de no poder salir del mapa.

# INTRODUCCIÓN

## Programación orientada a objetos como enfoque de desarrollo

Es fundamental comprender la metodología de desarrollo utilizada en este proyecto de investigación.

La programación orientada a objetos (POO) es un paradigma de programación que se basa en la representación de entidades del mundo real como objetos, los cuales tienen propiedades (atributos) y comportamientos (métodos). La POO se utiliza ampliamente en el desarrollo de software debido a su capacidad para estructurar el código de manera modular y reutilizable.

En el contexto de este proyecto de investigación, la programación orientada a objetos se aplicará para desarrollar el juego de disparos hacia jefes finales. Cada elemento del juego, como el jugador, los jefes finales, los disparos y los elementos del escenario, se representará como objetos individuales con sus propias características y comportamientos específicos.

Un beneficio clave de la POO es la capacidad de encapsular datos y funcionalidades relacionadas en objetos. Esto permite organizar y estructurar el código de manera más clara y comprensible. Por ejemplo, el objeto del jugador podría tener atributos como la posición en el mapa, la cantidad de vidas restantes y las habilidades especiales, así como métodos para moverse, disparar y colisionar con otros objetos.

Además de la organización, la POO también fomenta la reutilización de código. Mediante la creación de clases y objetos, es posible definir plantillas genéricas que luego pueden ser instanciadas para crear múltiples objetos con propiedades y comportamientos similares. En el contexto de este proyecto, esto significa que se pueden definir clases base para los jefes finales, los disparos y otros elementos del juego, y luego extender y personalizar esas clases según sea necesario.

Otro aspecto importante de la POO es la capacidad de establecer relaciones entre objetos mediante la herencia y la composición. La herencia permite crear jerarquías de clases, donde una clase más específica puede heredar atributos y comportamientos de una clase base más general. Por ejemplo, se podría tener una clase base para todos los enemigos, y luego clases específicas para cada jefe final con atributos y comportamientos adicionales.

## El esqueleto de "Galaga" como punto de partida

Se centra en la elección y utilización del juego "Galaga" como base para el desarrollo de este proyecto de investigación. Aquí se explorará cómo se utilizará el esqueleto del juego "Galaga" y cómo se adaptará y personalizará para lograr el resultado final deseado.

El juego "Galaga" es un clásico de los videojuegos de disparos lanzado por Namco en 1981. Es conocido por su estilo arcade y su jugabilidad adictiva, en la que los jugadores controlan una nave espacial y deben enfrentarse a oleadas de enemigos alienígenas.

En este proyecto de investigación, el esqueleto del juego "Galaga" se utilizará como punto de partida para el desarrollo del juego de disparos hacia jefes finales. Esto significa que se tomará la estructura básica y las funcionalidades del juego "Galaga" y se adaptarán y personalizarán para lograr el resultado final deseado, que es un juego similar a Cuphead.

El esqueleto del juego "Galaga" proporciona una base sólida para comenzar el desarrollo, ya que ya cuenta con muchas funcionalidades básicas implementadas. Estas funcionalidades incluyen el movimiento de la nave espacial, la generación y el movimiento del enemigo y los efectos de sonido.

Sin embargo, a medida que avance el desarrollo del proyecto, será necesario adaptar y personalizar el esqueleto de "Galaga" para cumplir con los requisitos y características específicas del juego de disparos hacia jefes finales. Por ejemplo, se deberá agregar la lógica y los comportamientos necesarios para implementar la lucha contra los jefes finales, los disparos aleatorios y las restricciones para evitar salir del mapa.

Además, es importante tener en cuenta que el objetivo final del proyecto es lograr un juego similar a Cuphead en términos de jugabilidad y estilo. Por lo tanto, a medida que se avance en el desarrollo, se realizarán modificaciones y mejoras para capturar la esencia y el desafío característico de Cuphead, adaptándolo a la estructura proporcionada por el esqueleto de "Galaga".

Utilizar el esqueleto de "Galaga" como punto de partida tiene varias ventajas. En primer lugar, permite ahorrar tiempo y esfuerzo al no tener que desarrollar desde cero todas las funcionalidades básicas del juego. En segundo lugar, brinda una estructura organizada y probada que puede servir como guía durante el desarrollo.

## Elementos clave del juego

Aspectos fundamentales y distintivos del juego de disparos hacia jefes finales.

Jefes Finales: Uno de los elementos clave del juego será la presencia de jefes finales desafiantes. Estos jefes serán enemigos poderosos y de gran tamaño que los jugadores deberán derrotar para avanzar en el juego.

Vidas Limitadas: Para aumentar la tensión y el desafío, el juego contará con un sistema de vidas limitadas. Los jugadores tendrán un número específico de vidas disponibles, y perderán una vida cada vez que sean golpeados por los ataques enemigos. Si los jugadores pierden todas sus vidas, el juego terminará. Esto enfatizará la importancia de la precisión y la evasión de los disparos enemigos.

Disparos Aleatorios: Otra característica del juego será la generación de disparos aleatorios por parte de los jefes finales.Estos disparos varían en patrón, lo que mantendrá a los jugadores en constante alerta y requerirá reflejos rápidos y precisión para evitar ser alcanzados.

Restricción de no poder salir del mapa: Una regla fundamental del juego será la restricción de no poder salir del mapa. Esto significa que los jugadores estarán confinados dentro de los límites del escenario y no podrán escapar o evadir a los jefes finales simplemente alejándose. Esto añadirá un elemento adicional de desafío estratégico, ya que los jugadores deberán moverse de manera inteligente y aprovechar las oportunidades para atacar y esquivar mientras están dentro del área de juego.

Estos elementos clave del juego trabajarán en conjunto para crear una experiencia desafiante y emocionante para los jugadores. La presencia de jefes finales, las vidas limitadas, los disparos aleatorios y la restricción del mapa contribuirán a la dificultad del juego y requerirán habilidades de juego estratégicas, reflejos rápidos y precisión por parte de los jugadores para tener éxito.

Al enfocarse en estos elementos clave, el juego de disparos hacia jefes finales se asemejará a Cuphead en términos de desafío y jugabilidad, brindando una experiencia similar pero con un enfoque propio y adaptado a partir del esqueleto de "Galaga".

# DESARROLLO

## Problema de pantalla principal

Durante el desarrollo de este proyecto, se encontró un problema relacionado con la pantalla principal. El inconveniente radicaba en que al intentar mostrar un nivel distinto al nivel 1, la pantalla se mostraba en blanco en lugar de renderizar el contenido correspondiente. Esta situación afectaba negativamente la jugabilidad y requería una solución adecuada.

Para solucionar este problema, se implementó un evento de acción y se realizó una modificación en el controlador principal. Además, fue necesario crear controladores individuales para cada nivel.

Mediante la inclusión del evento de acción en el controlador principal, se estableció una lógica que permitía mostrar correctamente la pantalla principal al cambiar de nivel. Al mismo tiempo, al crear controladores específicos para cada nivel, se aseguró una gestión adecuada de los elementos visuales y funcionales en cada etapa del juego.

Gracias a esta solución implementada, se logró superar el problema de la pantalla principal en blanco cuando se intentaba mostrar un nivel distinto al nivel 1. Como resultado, los jugadores pueden disfrutar de una experiencia de juego coherente y sin interrupciones visuales, sin importar en qué nivel se encuentren.

La resolución de este problema destaca la importancia de abordar los desafíos durante el desarrollo del proyecto y encontrar soluciones efectivas. Al solucionar el problema de la pantalla principal en blanco, se ha logrado mejorar significativamente la calidad visual y la experiencia general del juego.

## Problema de que cada nivel tenga sus propias imágenes y velocidad.

Durante el proceso de desarrollo de este proyecto, nos encontramos con un desafío significativo relacionado con la implementación de imágenes y velocidades únicas para cada nivel del juego. Esta cuestión planteó diversos problemas que requerían una solución detallada y cuidadosa.

Inicialmente, al tener diferentes clases para cada nivel, surgió la necesidad de establecer un método individual dentro del controlador principal para cada nivel en particular. Estos métodos desempeñaban un papel fundamental al asignar las imágenes específicas correspondientes a cada nivel. De esta manera, nos aseguramos de que cada nivel contará con sus propias imágenes distintivas.

Además, estos métodos se utilizaron en los eventos, como los botones, para garantizar que se activaran las imágenes correctas al seleccionar un nivel en particular. Esta integración entre los métodos de nivel y los eventos nos permitió establecer una conexión efectiva entre la interacción del jugador y la visualización de las imágenes correspondientes.

Por otro lado, encontramos un desafío adicional relacionado con el movimiento de las imágenes en el juego. Inicialmente, las imágenes se movían de arriba hacia abajo según el eje y, lo cual no se ajustaba a la lógica del juego base que estábamos utilizando como referencia, el "Galaga". Para solucionar esto, realizamos un cambio importante en la variable de coordenadas utilizada para el movimiento.

En lugar de utilizar la variable "yy" para controlar el movimiento vertical, decidimos emplear la variable "xx". Esto se debió a que el juego "Galaga" presentaba un movimiento de arriba hacia abajo y de izquierda a derecha. Al cambiar la variable de coordenadas, pudimos adaptar de manera efectiva el movimiento de las imágenes al comportamiento deseado en nuestro proyecto.

Además, para asegurarnos de que no hubiera espacios vacíos entre las repeticiones del ciclo de la imagen, implementamos una configuración específica para la variable "xx". Esta configuración consistía en inicializar "xx" con el ancho de la imagen correspondiente. De esta forma, logramos una transición suave y sin interrupciones visuales al repetir el ciclo de las imágenes.

En resumen, enfrentamos desafíos importantes en relación con la asignación de imágenes y velocidades únicas para cada nivel del juego. Estos desafíos fueron abordados mediante la creación de métodos individuales para cada nivel dentro del controlador principal. Además, ajustamos el movimiento de las imágenes utilizando la variable "xx" en lugar de "yy" para adaptarlo al comportamiento deseado del juego. Implementamos una configuración específica para asegurarnos de que el ciclo de las imágenes fuera fluido y sin espacios vacíos. Estas soluciones nos permitieron lograr un juego con niveles distintos, cada uno con sus propias imágenes y velocidades personalizadas.

## Adaptar el disparo de "galaga" a "cuphead"

Durante el proceso de adaptación del disparo de "Galaga" al estilo de juego de "Cuphead", nos encontramos con un desafío técnico que requería una solución detallada y precisa.

Inicialmente, al intentar implementar el código de disparo, nos dimos cuenta de que solo se generaba un único disparo en lugar de los múltiples cuadritos que caracterizaban el juego original de "Cuphead". Esta limitación no encajaba con nuestra visión de recrear fielmente la dinámica de combate intensa y frenética de "Cuphead", lo cual nos motivó a buscar una solución adecuada.

Para abordar este problema, nos centramos en la declaración del disparo y optamos por utilizar un ArrayList en el código. La ventaja de utilizar un ArrayList radicaba en su capacidad para almacenar múltiples objetos, en este caso, los disparos. Al hacer esto, pudimos generar disparos adicionales cada vez que se presionaba el botón correspondiente, lo cual era esencial para capturar la esencia del juego original.

La implementación del ArrayList nos permitió lograr un comportamiento más acorde con "Cuphead", donde los jugadores pueden disparar rápidamente y enfrentar múltiples desafíos en pantalla. Esto añadió un nivel de intensidad y emoción al juego, manteniendo la fidelidad con la experiencia de juego original que queríamos recrear.

Es importante destacar que si hubiéramos mantenido únicamente un disparo único, el ritmo y la diversión del juego se verían comprometidos. Al permitir que se generen múltiples disparos mediante el uso del ArrayList, conseguimos mantener la jugabilidad fluida y enérgica de "Cuphead". Esta decisión nos ayudó a evitar que el juego se volviera monótono y a garantizar que los jugadores pudieran enfrentar desafíos emocionantes y variados.

## El problema con los disparos y movimiento de los jefes

Durante el desarrollo de nuestro proyecto, nos encontramos con desafíos significativos en relación a los disparos y el movimiento de los jefes en el juego.

Uno de los desafíos fue implementar un movimiento autónomo para los jefes. Inicialmente, nos enfrentamos a la dificultad de encontrar una forma adecuada de hacer que los jefes se movieran por sí mismos de manera fluida y realista en el escenario. Para resolver esta situación, investigamos y exploramos diferentes enfoques. Finalmente, encontramos una solución utilizando la biblioteca "random" de Java, específicamente haciendo uso de la función de esta biblioteca. Esta función nos permitió generar movimientos aleatorios para los jefes, lo cual agregó un elemento de imprevisibilidad y desafío al juego. Al incorporar este movimiento autónomo, logramos que los jefes se desplazarán de manera más dinámica y generarán una experiencia de juego más emocionante.

Por otro lado, nos encontramos con el desafío de asignar imágenes adecuadas a los disparos en cada nivel. Era importante que los disparos no fueran simplemente cuadritos transparentes, sino que tuvieran dimensiones y aspectos visuales distintos que se adaptaran a la temática de cada nivel. Ante este desafío, buscamos orientación y asesoramiento adicional de nuestro profesor, quien nos sugirió una estrategia efectiva. Siguiendo su recomendación, decidimos tratar las imágenes de los disparos como si fueran personajes adicionales en el juego. Esto nos permitió declarar y gestionar estas imágenes dentro de un bucle "for", asignando diferentes dimensiones y características visuales a los disparos en cada nivel. De esta manera, logramos enriquecer la experiencia visual del juego y asegurarnos de que cada nivel tuviera una identidad única.

Al utilizar la biblioteca para implementar un movimiento autónomo de los jefes, pudimos agregar variedad y emoción al juego. Además, al tratar las imágenes de los disparos como personajes adicionales y asignarles dimensiones y características específicas en cada nivel, logramos mejorar significativamente la calidad visual y la inmersión en el juego. Estas soluciones nos permitieron superar los obstáculos encontrados y alcanzar el resultado deseado en nuestro proyecto de desarrollo de videojuegos.

## Colisión de disparos

Durante el desarrollo de nuestro juego, nos encontramos con un desafío importante relacionado con la colisión de disparos. Al implementar la funcionalidad de colisión, nos enfrentamos a un error persistente debido a una variable que se encontraba en estado nulo. Esto generaba una excepción de clase y dificultaba el correcto funcionamiento de la detección de colisiones.

Para abordar este problema, investigamos y analizamos detenidamente el código relevante. Nos dimos cuenta de que necesitábamos acceder al rectángulo del personaje y del jefe final para llevar a cabo la detección de colisiones de manera precisa. Sin embargo, la variable necesaria para obtener estos rectángulos no estaba inicializada adecuadamente, lo que causaba el error mencionado anteriormente.

Para solucionar esta situación, decidimos modificar el enfoque y recurrir a una solución basada en la comunicación entre clases. Implementamos un método en el controlador principal que recibía como parámetro la instancia de la clase correspondiente al personaje y al jefe final. Al hacerlo, logramos obtener acceso a la variable necesaria y evitar el problema de la variable nula.

Con la instancia de la clase del personaje y del jefe final disponible, pudimos utilizar los rectángulos asociados a cada uno de ellos en el proceso de detección de colisiones. Esto permitió que los disparos interactuaran de manera adecuada con los elementos del juego, como el personaje controlado por el jugador y los jefes finales. En esencia, estábamos pidiendo a estas clases que nos proporcionarán la información necesaria para llevar a cabo la detección de colisiones de manera efectiva.

Es importante destacar que nos basamos en el código existente relacionado con la colisión del juego "Space Invader", adaptándolo y aplicándolo a nuestro proyecto. Esta referencia nos sirvió como punto de partida para abordar el problema y encontrar una solución sólida.

El desafío de implementar la colisión de disparos nos llevó a enfrentar un error de variable nula. Para solucionarlo, modificamos nuestro enfoque y establecimos una comunicación entre clases, solicitando la información necesaria al personaje y al jefe final mediante un método en el controlador principal. Al obtener acceso a los rectángulos correspondientes, logramos realizar una detección de colisiones precisa y garantizar un comportamiento adecuado en nuestro juego. Aprovechamos el código existente del juego "Space Invader" como referencia, adaptándolo a nuestras necesidades específicas. Este enfoque nos permitió superar el desafío y avanzar en el desarrollo de nuestro juego de manera exitosa.

## Cambio de música y animación al morir.

Durante el desarrollo del juego, nos enfrentamos a un desafío relacionado con el cambio de música y animación al ocurrir la muerte del jugador. Inicialmente, teníamos implementada una funcionalidad que modificaba tanto la música como la imagen al producirse la muerte del personaje. Sin embargo, nos encontramos con un error persistente en el que la variable asociada a esta funcionalidad se encontraba en estado nulo.

Para solucionar este problema, decidimos realizar algunos ajustes. En primer lugar, eliminamos la reproducción de la música al ocurrir la muerte y nos enfocamos únicamente en la animación. Esto nos permitió simplificar el código y evitar posibles problemas relacionados con la variable nula.

Para lograr el cambio de animación al morir, implementamos un método que contaba el número de vidas restantes del jugador. Cuando esta cantidad llegaba a cero, se activaba la animación correspondiente a la muerte. Este enfoque nos permitió asegurarnos de que la animación se ejecutara adecuadamente al momento de la muerte del personaje.

Si bien inicialmente teníamos la intención de cambiar tanto la música como la imagen al morir, decidimos centrarnos en la animación debido a las limitaciones encontradas con la variable nula. Aunque no logramos incorporar el cambio de música como se había planeado inicialmente, pudimos garantizar una transición visual efectiva y coherente al momento de la muerte del personaje.

Nos encontramos con un problema en la implementación del cambio de música y animación al morir en el juego. Debido a una variable nula, nos vimos obligados a ajustar nuestro enfoque y nos enfocamos exclusivamente en la animación de la muerte del personaje. Implementamos un método que contaba las vidas restantes del jugador y activaba la animación correspondiente cuando estas llegaban a cero. Aunque no logramos incorporar el cambio de música como se había planeado inicialmente, conseguimos asegurar una transición visual efectiva en el juego.

## Problemas de botón de los niveles para regresar a pantalla principal

Durante el desarrollo del proyecto, nos encontramos con diversos problemas relacionados con el botón destinado a regresar a la pantalla principal desde los niveles del juego. Estos problemas generaban una serie de bugs y fallos en la funcionalidad del juego que afectaban negativamente la experiencia del jugador.

Uno de los problemas identificados era la reproducción repetitiva de la música al utilizar el botón para regresar a la pantalla principal. Cada vez que se presionaba el botón, la música se volvía a reproducir desde el principio, lo cual generaba una experiencia auditiva incómoda y poco deseable. Esto se debía a una falta de manejo adecuado de los eventos de música al salir de los niveles.

Además, se observaron fallos en el renderizado de los niveles al utilizar el botón. En algunos casos, al regresar a la pantalla principal, el nivel no se pintaba correctamente o se mostraba en blanco. Esto se debía a una interacción incorrecta entre los controladores y la gestión de los gráficos al cambiar entre las diferentes escenas del juego.

Otro problema identificado estaba relacionado con la movilidad del personaje controlado por el jugador. Al utilizar el botón para regresar a la pantalla principal, en algunos casos el personaje dejaba de moverse correctamente. Esto se debía a errores en la lógica de control y al manejo de eventos que se activaban al cambiar de escena.

Por último, se encontraron problemas con las colisiones entre los elementos del juego. Al utilizar el botón de regreso, las colisiones entre el personaje y otros objetos del juego dejaban de funcionar adecuadamente. Esto llevaba a situaciones en las que el personaje podía atravesar obstáculos o enemigos sin que se aplicarán las reglas de colisión establecidas.

Ante la persistencia de estos bugs y la complejidad para resolverlos, se tomó la decisión de eliminar por completo el botón de regreso a la pantalla principal desde los niveles. En su lugar, se implementó una lógica que redireccionaba automáticamente al jugador al menú principal al finalizar las vidas del jugador o del jefe final. Esta solución se consideró como la más adecuada para evitar problemas innecesarios y mantener la estabilidad del juego.

La eliminación del botón de regreso simplificó el flujo del juego y evitó la aparición de los problemas mencionados. Al redirigir automáticamente al jugador al menú principal al finalizar las vidas, se evitó la interacción problemática con los controladores, eventos y elementos visuales. Esta solución permitió disfrutar de un juego más estable y sin las complicaciones que surgían al utilizar el botón de regreso.

En resumen, los problemas relacionados con el botón de regreso a la pantalla principal desde los niveles fueron abordados mediante la eliminación del botón y la implementación de una redirección automática al menú principal. Esta solución resolvió los errores y fallos observados, mejorando significativamente la experiencia de juego al evitar complicaciones innecesarias y mantener la estabilidad del juego en general.

## UML

# 

# 

# 

# 

# 

# 

# 

# 

# 

# CONCLUSIONES

Durante las etapas iniciales de nuestro proyecto, nos encontramos con una serie de problemas que desencadenaron más complicaciones. Esta cadena de errores nos obligó a abordar cada desafío de manera individual, resolviéndolos uno a uno. Sin embargo, contábamos con una ventaja fundamental: comenzamos con una base de otro juego similar al que estábamos desarrollando. Esta base nos brindó una guía clara para orientarnos en nuestro proyecto, evitando comenzar desde cero sin saber por dónde empezar.  
El desarrollo de nuestro proyecto de creación de un juego de disparos hacia jefes finales utilizando la lógica tipo Atari, la programación orientada a objetos y el esqueleto del juego "Galaga" nos llevó a enfrentar diversos desafíos que requerían soluciones innovadoras. Las colisiones, los disparos y las animaciones fueron aspectos críticos que presentaron obstáculos significativos en el camino.

En relación a las colisiones, superamos problemas de variables nulas y excepciones mediante la implementación de métodos que permitían obtener los rectángulos de colisión de los personajes y jefes. Esto nos permitió detectar de manera precisa las colisiones y aplicar las acciones correspondientes en el juego.

En cuanto a los disparos, inicialmente nos encontramos con limitaciones en el código existente que solo permitía un único disparo en pantalla. Para solucionar esto, utilizamos un enfoque basado en la declaración de un ArrayList, lo que nos permitió generar múltiples instancias de disparo y lograr una experiencia más fiel al juego original, agregando mayor dinamismo y desafío.

En relación a las animaciones, enfrentamos desafíos al cambiar la música y la imagen al momento de la muerte del personaje. Mediante la implementación de un método que verificaba las vidas restantes, logramos activar la animación correspondiente al llegar a cero vidas. Esto implicó eliminar la música y enfocarnos en la imagen, lo que nos permitió superar los errores relacionados con variables nulas y obtener una transición fluida al morir en el juego.

A lo largo de este proceso, aprendimos a enfrentar y resolver problemas complejos, lo cual nos permitió adquirir un conocimiento más profundo sobre la programación de videojuegos y mejorar nuestras habilidades técnicas. La colaboración entre miembros del equipo también fue un factor clave, ya que nos permitió abordar los desafíos desde diferentes perspectivas y encontrar soluciones efectivas.

En definitiva, a pesar de los obstáculos encontrados, logramos desarrollar un juego de disparos hacia jefes finales con una lógica tipo Atari, utilizando la programación orientada a objetos y el esqueleto del juego "Galaga". El proceso de solucionar los problemas relacionados con las colisiones, los disparos y las animaciones nos permitió obtener un juego funcional y satisfactorio, y nos brindó una valiosa experiencia en el desarrollo de videojuegos.

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# BIBLIOGRAFÍA

<https://www.spriters-resource.com/pc_computer/cupheaddontdealwiththedevil/sheet/161864/>

github