Facultad de ingeniería, arquitectura y diseño

Fecha: 2 de abril del 2024

Integrantes:
Pedro Carlos Balderrama Campos

Metodología de la Investigación
Profesor: Carlos Saúl López Sánchez

Proyecto de Investigación sobre el desarrollo de videojuegos

ÍNDICE

Antecedentes del Desarrollo de Software

El desarrollo de software tiene una historia rica y diversa que se remonta a los primeros días de la informática, evolucionando significativamente a lo largo de las décadas. A continuación, se presenta un resumen de algunos antecedentes clave en el desarrollo de software.

Los primeros días (1940s-1950s)

La historia del desarrollo de software comenzó con la invención de las primeras computadoras programables. Durante esta época, el software se desarrollaba principalmente en lenguaje de máquina o en lenguajes de bajo nivel como el ensamblador, lo que requería una comprensión detallada del hardware subyacente.

Introducción de lenguajes de alto nivel (1950s-1960s)

La creación de lenguajes de programación de alto nivel como FORTRAN (1957), LISP (1958), y COBOL (1959) marcó un cambio significativo. Estos lenguajes permitieron a los programadores escribir código de manera más abstracta, sin necesidad de conocer los detalles del hardware.

Desarrollo estructurado y metodologías (1970s)

La década de 1970 vio el nacimiento del desarrollo de software estructurado, que enfatizaba la importancia de la planificación, el diseño, y el testing en el proceso de desarrollo. Durante este tiempo, se introdujeron metodologías como el Modelo en Cascada.

Programación orientada a objetos (1980s-1990s)

La programación orientada a objetos (OOP) surgió como un paradigma dominante, con lenguajes como SmallTalk, C++, y Java liderando el camino. Este enfoque promovía la reutilización de código y un diseño de software más modular.

Desarrollo ágil (finales de 1990s-presente)

En respuesta a las limitaciones de las metodologías de desarrollo más tradicionales, se introdujo el desarrollo ágil. Se centraba en la flexibilidad, la entrega continua, y la colaboración estrecha entre los equipos de desarrollo y los stakeholders. El Manifiesto Ágil, publicado en 2001, formalizó estos principios.

Avances recientes

En las últimas décadas, hemos visto avances significativos en áreas como el desarrollo de software basado en componentes, servicios web, programación funcional, y DevOps. Además, la proliferación de la computación en la nube y el desarrollo de aplicaciones móviles han transformado el panorama del desarrollo de software.

Estos antecedentes muestran cómo el desarrollo de software ha evolucionado de ser una práctica altamente técnica y especializada a una disciplina más colaborativa y orientada al diseño, con una amplia gama de metodologías y enfoques disponibles para adaptarse a diferentes necesidades y contextos de proyecto.

Antecedentes del Desarrollo de VideoJuegos

La evolución del desarrollo de software ha influido profundamente en el desarrollo de videojuegos, un campo que combina creatividad artística con innovaciones tecnológicas. Un Un antecedente relevante del desarrollo de videojuegos que se destaca dentro de esta historia es la" Programación Orientada a Objetos (POO)" durante las décadas de 1980 y 1990. Este paradigma de programación promovió la reutilización de código y un diseño de software más modular, aspectos críticos en el desarrollo de videojuegos.

En los inicios del desarrollo de videojuegos, los programadores enfrentaban desafíos significativos debido a las limitaciones de hardware y la necesidad de optimizar recursos al máximo. Con la adopción de la POO, los desarrolladores de videojuegos encontraron un marco que les permitía construir juegos más complejos y ricos en características de manera eficiente.

Lenguajes de programación como C++ y Java, que soportan la POO, se convirtieron en herramientas fundamentales para crear motores de juego reutilizables y componentes de software que podían ser compartidos entre diferentes proyectos.

La OOP facilitó el manejo de la complejidad inherente al desarrollo de videojuegos, permitiendo a los desarrolladores centrarse más en la creatividad y menos en los detalles de bajo nivel del hardware. Este enfoque modular y reutilizable aceleró el proceso de desarrollo, permitiendo iteraciones más rápidas y la experimentación con nuevas ideas de gameplay y mecánicas de juego.

Este antecedente no solo marcó una evolución tecnológica, sino que también propició una transformación cultural en el desarrollo de videojuegos, promoviendo prácticas colaborativas y una mayor experimentación, lo cual ha sido fundamental para el crecimiento y la diversificación del sector de los videojuegos.

Desarrollo del juego tetris

El desarrollo de Tetris, uno de los videojuegos más icónicos de la historia, se remonta a junio de 1984 en la Unión Soviética, específicamente en Moscú. Fue creado por Alexey Pajitnov, un ingeniero de software que trabajaba en el Centro de Computación Dorodnitsyn de la Academia de Ciencias de la Unión Soviética. Inspirado por su afición a los puzzles, Pajitnov se propuso desarrollar un juego que combinara elementos de puzle con una mecánica de juego dinámica. La primera versión de Tetris fue programada en un Elektronika 60, un ordenador muy limitado en capacidades gráficas, por lo que el juego original no incluía los coloridos bloques que caracterizan a las versiones posteriores. En su lugar, utilizaba caracteres ASCII para representar los Tetriminos, las formas geométricas compuestas de cuatro cuadros que caen en el campo de juego. Pajitnov diseñó Tetris con la idea de que completar líneas con los Tetriminos sin dejar espacios vacíos sería tanto desafiante como satisfactorio para el jugador. A medida que el jugador completaba líneas, estas desaparecían, liberando espacio para más Tetriminos, y el juego se aceleraba, aumentando la dificultad.

El nombre Tetris; deriva de la palabra griega “tetra” ; (que significa “cuatro”;), refiriéndose al número de cuadrados que componen cada pieza en el juego, combinado con la afición de Pajitnov por el tenis. La simplicidad, junto con la profunda complejidad y el aspecto adictivo de Tetris, lo convirtieron rápidamente en un fenómeno. A pesar de las barreras políticas y las dificultades iniciales para comercializar el juego fuera de la Unión Soviética, Tetris se difundió a nivel mundial, encontrando su camino hacia diversas plataformas de juego y convirtiéndose en uno de los videojuegos más vendidos y reconocidos de todos los tiempos. La historia de Tetris es un testimonio del poder de la creatividad en el desarrollo de videojuegos y de cómo un concepto simple puede convertirse en un fenómeno cultural global.

Introducción

El mundo del desarrollo de software ha sido el pilar sobre el que se han construido las tecnologías que hoy día conforman la esencia de nuestra vida cotidiana. Desde los primeros días de la informática, marcados por la creación de las primeras computadoras programables, hasta la era actual dominada por la computación en la nube, las aplicaciones móviles y más recientemente, la inteligencia artificial; la evolución del desarrollo de software ha sido constante y profundamente impactante. Esta evolución no solo ha transformado la manera en que interactuamos con las tecnologías, sino que también ha redefinido los límites de lo que es posible en términos de innovación. La historia del desarrollo de software es evidencia de la superación humana, donde cada avance ha sido un paso hacia niveles más altos de eficiencia y creatividad.

Además, la programación orientada a objetos y el desarrollo ágil han remodelado profundamente el panorama del desarrollo de software, introduciendo conceptos como la reutilización de código, el diseño modular y la colaboración estrecha entre equipos. Estas metodologías no solo han mejorado la calidad y la eficiencia del software desarrollado, sino que también han fomentado una cultura de experimentación y adaptabilidad, esencial para mantenerse al ritmo de los rápidos cambios tecnológicos.

En el contexto del desarrollo de videojuegos, estos avances han sido especialmente cruciales. La programación orientada a objetos, por ejemplo, ha permitido a los desarrolladores de videojuegos enfrentar y superar los desafíos únicos de este campo, como la gestión de complejas interacciones de juego y la representación de mundos ricos y dinámicos.

Objetivo

El objetivo de esta investigación es desarrollar un juego tipo Arcade en lenguaje C++, con el fin de ofrecer una pequeña colección de juegos entretenidos. Este proyecto no solo busca rendir homenaje a la era dorada de los videojuegos Arcade, caracterizada por su simplicidad y atractivo universal, sino que también pretende ser un ejercicio práctico en la aplicación de las técnicas y principios del desarrollo de software moderno. A través de este proyecto, se explorarán conceptos como la programación orientada a objetos para el diseño de un sistema de juego modular y reutilizable, así como las prácticas de desarrollo ágil para gestionar el proceso de creación del juego de manera eficiente y adaptativa.

El desarrollo de este juego tipo Arcade en C++ servirá como un caso de estudio concreto sobre cómo las metodologías y avances tecnológicos en el desarrollo de software pueden ser aplicados en el contexto específico del desarrollo de videojuegos.

Objetivo específico

El objetivo específico del proyecto es desarrollar un juego educativo en lenguaje C++ (apoyándome con la librería diseñada para crear videojuegos llamada “raylib”) que refuerce las habilidades matemáticas básicas de los usuarios, enfocándose en operaciones como sumas, restas y multiplicaciones. El propósito principal es crear un juego que sea divertido y educativo, facilitando el aprendizaje de las matemáticas a través de mecánicas de juego atractivas y desafiantes. Se pondrá especial énfasis en crear una interfaz intuitiva y una jugabilidad fluida, permitiendo a los usuarios sumergirse completamente en la experiencia de aprendizaje. Además, se buscará implementar un proceso de desarrollo ágil que asegure la rápida entrega de nuevas características y permita iteraciones constantes basadas en el feedback de los usuarios, con el fin de optimizar continuamente la experiencia del juego y satisfacer las necesidades educativas del público objetivo.

Justificación

El desarrollo de un juego educativo en lenguaje C++ que refuerce las habilidades matemáticas básicas de los usuarios se justifica por su capacidad para proporcionar una experiencia de aprendizaje divertida y efectiva. Al considerar diversos aspectos, como la investigación previa en el campo de la educación, la consulta con expertos en pedagogía y la evaluación del mercado de juegos educativos, se garantiza un proyecto respaldado por un sólido fundamento de conocimiento y orientación profesional. Este enfoque también asegura que el juego sea relevante y atractivo para los usuarios, promoviendo el aprendizaje de las matemáticas de una manera lúdica y motivadora, lo que contribuye a su viabilidad y éxito a largo plazo. Además, el continuo refinamiento del juego para alinearlo con metas educativas personales y profesionales permite la creación de un producto final enriquecedor y eficaz, lo que promueve una experiencia positiva tanto para los estudiantes como para los desarrolladores.

Aspectos Técnicos del Ingeniero

Para desarrollar "MathQuest", empleamos Visual Studio Code, un editor de código versátil y gratuito compatible con Windows, macOS y Linux. Este editor, altamente personalizable con miles de extensiones, facilita la programación en C++, nuestro lenguaje de elección debido a su eficiencia en la gestión de gráficos y rendimiento. Utilizamos la biblioteca raylib para manejar gráficos y eventos. Git nos ayuda a mantener el control de versiones, permitiendo una colaboración efectiva y segura en el equipo. Seguimos metodologías ágiles como Scrum para organizar nuestro trabajo en sprints, lo que nos permite iterar rápidamente y adaptarnos al feedback. La depuración se realiza con las herramientas de Visual Studio Code, garantizando un juego estable. La arquitectura del juego se estructura utilizando patrones de diseño como el patrón de fábrica y el patrón observador, lo que permite un código modular y fácil de mantener.

Aspectos Técnicos del Usuario

"MathQuest" está diseñado para ser accesible y atractivo para los usuarios. La interfaz es intuitiva y fácil de navegar, con controles simples que facilitan la interacción. Los efectos visuales y sonoros proporcionan una retroalimentación inmediata que mejora la experiencia de juego. El juego es compatible con Windows, macOS y Linux, y requiere una tarjeta gráfica adecuada y al menos 12 MB de espacio en disco duro para funcionar sin problemas. Hemos optimizado el juego para asegurar una jugabilidad fluida y hemos incluido varios niveles de dificultad para adaptarse a todos los jugadores. Es importante que el sistema esté actualizado y que se utilice la versión correcta de la biblioteca raylib para aprovechar al máximo las características del juego.

Parte Técnica del Usuario

Para que los usuarios puedan disfrutar plenamente de "MathQuest", es necesario que tengan conocimientos básicos de aritmética, ya que el juego se centra en operaciones matemáticas fundamentales como la suma, resta, multiplicación y división. Además, los usuarios necesitan una configuración de hardware adecuada: una tarjeta gráfica compatible y al menos 12 MB de espacio en disco duro para asegurar un funcionamiento fluido. El proceso de instalación del juego es sencillo, con un instalador que guía al usuario paso a paso. Una vez instalado, los usuarios pueden ajustar parámetros gráficos, controles y sonido desde un menú de configuración según sus preferencias. Además, proporcionamos un manual del usuario con

instrucciones detalladas sobre cómo jugar, descripciones de las mecánicas del juego y soluciones a problemas comunes. Para el soporte técnico, ofrecemos una sección de preguntas frecuentes (FAQ) y un foro en línea donde los usuarios pueden resolver sus dudas y problemas. Esto asegura que los usuarios tengan una experiencia de juego satisfactoria y sin complicaciones.

Conclusión

El desarrollo de software ha pasado de la programación en lenguajes de máquina y ensamblador en los primeros días de la informática, a la adopción de lenguajes de alto nivel y metodologías estructuradas en las décadas posteriores. La programación orientada a objetos en los 1980s y 1990s, junto con el desarrollo ágil desde finales de los 1990s, ha mejorado significativamente la eficiencia, modularidad y flexibilidad en el desarrollo de software.

En el ámbito de los videojuegos, estos avances han sido esenciales para manejar la complejidad y promover la creatividad. El desarrollo de "MathQuest" busca aprovechar estos principios, utilizando C++ y técnicas modernas de desarrollo para crear un juego educativo tipo Arcade. "MathQuest" no solo pretende ofrecer una experiencia de juego entretenida y cautivadora, sino también ser un ejemplo práctico de cómo aplicar metodologías de desarrollo de software avanzadas en un contexto lúdico y educativo.