Instituto Tecnológico de Costa Rica

Unidad de Computación

Trivia Quirk

Integrantes del equipo:
Esteban Rodríguez Salas
Luis Andrés Mendez Campos
Andrés Quiros Rojas

Sede San Carlos

16.11.2023

Introducción

El proyecto en cuestión consiste en un tradicional videojuego de Trivia sobre preguntas divididas por categorías que el jugador en cuestión podrá seleccionar, dicho juego contará con un mínimo de 30 preguntas por partida y contará con pantalla de inicio, menú, juego interactivo y pantalla de resultados finales al completar las 30 preguntas.

La naturaleza del proyecto resulta sumamente provechosa desde el punto de vista del curso debido a la necesidad de utilizar una interfaz gráfica para todo lo que respecta el funcionamiento del proyecto por lo que genera una enseñanza de manera productiva, además el hecho de aplicar las distintas estrategías dentro de la programación orientada a objetos de cara a conseguir un resultado exitoso como el manejo eficiente entre clases y paquetes mejorando las técnicas de programación modular.

Se tiene la necesidad de hacer un juego completamente funcional, es por eso que el proyecto contará con el crucial requerimiento de mantener completamente controlado el manejo de excepciones de manera que los usuarios que gocen del resultado final puedan tener una experiencia completamente fluida sin ningún tipo de caída consiguiendo así mejorar nuestra capacidad de realizar software funcional.

Análisis del problema

Necesidad de una Interfaz Gráfica Atractiva y Funcional:

- **Problema:** Diseñar una interfaz gráfica que sea intuitiva y atractiva para los usuarios, manteniendo la funcionalidad necesaria para una experiencia de juego fluida.
- **Relevancia:** Una interfaz mal diseñada podría resultar en una experiencia de usuario pobre, disminuyendo el interés en el juego.

Desarrollo de un Sistema de Preguntas Eficiente:

- **Problema**: Crear un sistema que gestione un amplio conjunto de preguntas, categorizándolas y presentándolas de manera aleatoria y equilibrada en cada partida.
- **Relevancia:** La falta de variedad o un equilibrio inadecuado en las preguntas podría hacer que el juego sea monótono o injustamente sesgado hacia ciertas categorías.

Manejo de Excepciones y Estabilidad del Software:

- **Problema:** Asegurar que el juego maneje adecuadamente las excepciones para evitar fallos y garantizar una experiencia de usuario continua.
- **Relevancia:** Los errores no manejados o fallos en el juego pueden frustrar a los usuarios y afectar negativamente la percepción del software.

Implementación de Funcionalidades Multijugador:

- **Problema:** Desarrollar un modo multijugador que sea justo, competitivo y que mantenga el interés de los jugadores.
- **Relevancia:** Un modo multijugador mal implementado puede resultar en una experiencia desequilibrada y poco atractiva para los usuarios.

Aplicación de Principios de Programación Orientada a Objetos:

- **Problema:** Utilizar eficientemente la programación orientada a objetos para facilitar la modularidad y mantenibilidad del código.
- **Relevancia:** Un código mal estructurado puede resultar en dificultades de mantenimiento y expansión del juego en el futuro.

Accesibilidad y Diversidad de Contenidos:

- **Problema:** Asegurarse de que el contenido del juego sea accesible y atractivo para un público diverso.
- **Relevancia:** Un juego que no considere la diversidad y accesibilidad puede limitar su alcance y no ser inclusivo para todos los usuarios potenciales.

Evaluación y Mejora Continua:

- **Problema:** Establecer un mecanismo para recoger retroalimentación y mejorar el juego continuamente.
- **Relevancia:** Sin un proceso de evaluación y mejora, el juego podría no evolucionar para satisfacer las expectativas cambiantes de los usuarios.

Solución del problema

Diseño de Interfaz Gráfica Atractiva y Funcional:

- **Solución:** Implementar principios de diseño de interfaz de usuario (UI) y experiencia de usuario (UX) para crear una interfaz intuitiva, visualmente atractiva y fácil de navegar. Herramientas como Sketch o Adobe XD pueden ser útiles para prototipos y pruebas de usabilidad.

Desarrollo de un Sistema de Preguntas Eficiente:

- **Solución:** Crear una base de datos robusta para las preguntas, asegurando una variedad y un balance adecuado. Implementar algoritmos que seleccionen preguntas de manera aleatoria, pero equitativa, en cada categoría.

Manejo de Excepciones y Estabilidad del Software:

- **Solución:** Realizar pruebas exhaustivas (unitarias, de integración, de sistema) para identificar y manejar excepciones. Implementar un manejo de errores robusto para asegurar que el juego permanezca estable y reactivo frente a situaciones inesperadas.

Implementación de Funcionalidades Multijugador:

- **Solución:** Diseñar un sistema de puntuación justo y un mecanismo de matchmaking equilibrado. Implementar funcionalidades de red que permitan una experiencia multijugador fluida y sin demoras.

Aplicación de Principios de Programación Orientada a Objetos:

- **Solución:** Utilizar técnicas como la encapsulación, la herencia y el polimorfismo para estructurar el código de manera modular y mantenible. Esto facilita futuras actualizaciones y escalabilidad del juego.

Accesibilidad y Diversidad de Contenidos:

- **Solución:** Asegurarse de que el juego sea accesible para diferentes grupos de usuarios, incluyendo la implementación de características para usuarios con discapacidades. Diversificar el contenido de las preguntas para abarcar un amplio espectro de temas e intereses.

Evaluación y Mejora Continua:

- **Solución:** Establecer un sistema de retroalimentación donde los jugadores puedan enviar sus opiniones y sugerencias. Utilizar esta información para realizar actualizaciones regulares y mejoras basadas en las necesidades y preferencias de los usuarios.

Análisis de resultados

Tarea/Requerimiento	Estado	Observaciones
Desarrollo de Interfaz Gráfica	**Completo**	
Sistema de Preguntas y Base de Datos	**Completo**	
Manejo de Errores y Excepciones	**Completo**	
Pruebas Unitarias y de Integración	**Completo**	
Modo Multijugador y Gestión de Jugadores	**Completo**	
Implementación de Patrones de Diseño	**Completo**	
Documentación Javadoc y Reportes	**Completo**	
Integración de Categoría de Preguntas Externa	**Completo**	
Modo Multijugador Avanzado con Sockets	**No Iniciado**	**Pendiente de inicio tras finalizar otras tareas prioritarias**
Control de Versiones y	**Completo**	**Se está utilizando Git.**

Entrega Final		

Conclusiones

El proyecto "TriviaQuirk" del Curso de Porgramacion Orientada a Objetos, es un destacado ejemplo de cómo la teoría y la práctica pueden combinarse armoniosamente en el campo de la enseñanza de la computación. Se ha demostrado una comprensión profunda y aplicada de conceptos clave en el desarrollo de software, como la interfaz gráfica de usuario, la programación orientada a objetos, y la importancia de un diseño inclusivo y accesible.

Este proyecto no solo ha logrado crear un juego interactivo y educativo que puede atraer a una amplia gama de usuarios, sino que también ha servido como un vehículo para que los estudiantes apliquen y profundicen su conocimiento en áreas técnicas esenciales. El énfasis en la estabilidad del software, la eficiencia en la gestión de bases de datos y la implementación de prácticas de programación avanzadas refleja un enfoque educativo que valora tanto la teoría como la aplicación práctica.

Además, el proyecto ha reconocido la importancia de la retroalimentación continua y la mejora, preparando a los estudiantes para el dinámico y siempre cambiante mundo del desarrollo de software. "TriviaQuirk" es, por lo tanto, más que un proyecto de clase; es un microcosmos del proceso de desarrollo de software en el mundo real, proporcionando a los estudiantes una experiencia invaluable que seguramente beneficiará sus futuras carreras en tecnología.

TriviaQuirk es un testimonio del valor de integrar el aprendizaje práctico en la educación en computación, preparando a los estudiantes no solo para superar desafíos técnicos, sino también para pensar de manera innovadora y creativa en el diseño y la implementación de soluciones de software.

Recomendaciones

- Delimitar ciertos requerimientos que estén relacionados con la implementación de archivos tipo interface y clases abstractas para obtener un producto mucho más eficiente.
- Utilizar de manera obligatoria ciertas guías relacionadas a la creación de interfaces gráficas de usuario profesionales para aplicarlas dentro del proyecto.
- Seleccionar de un color verde la respuesta correcta en caso de seleccionar una respuesta que haya sido incorrecta.
- Dentro de las partidas en modo multijugador implementar un sistema de puntos visible para ambos jugadores los cuales se van restando o sumando en función del desempeño de los jugadores.

Referencias

Baeldung, & Baeldung. (2023). A guide to Java sockets | Baeldung. Baeldung. https://www.baeldung.com/a-guide-to-java-sockets

Java Color Codes - JavatPoint. (s. f.). www.javatpoint.com. https://www.javatpoint.com/java-color-codes

ChuWiki. (2023). Serialización de objetos en java - ChUWiki. ChuWiki. https://chuidiang.org/index.php?title=Serializaci%C3%B3n_de_objetos_en_java

Sentencias bucle en Java. (2023, 6 octubre). Manual Web. https://www.manualweb.net/java/sentencias-bucle-java/

González, J. D. M. (2021). Sockets (Cliente-Servidor). www.programarya.com. https://www.programarya.com/Cursos-Avanzados/Java/Sockets

GeeksforGeeks. (2023). Socket programming in Java. GeeksforGeeks. https://www.geeksforgeeks.org/socket-programming-in-java