Universidad Austral de Chile

Conocimiento y Naturaleza

Proyecto: Juego de Buscaminas

INFO 229 Arquitectura de Software

Integrantes:

- Katherine Zapata
- Fabián Ruiz
- Sebastián Vásquez

Introducción

La arquitectura de software es el diseño planificado de un sistema informático el cual se define cómo los componentes de un sistema que interactúan y se relacionan para cumplir con los objetivos funcionales y no funcionales del software.

Desarrollaremos una versión digital del clásico juego de buscaminas, utilizando Python y la biblioteca Tkinter para la interfaz gráfica de usuario. El juego está diseñado para ser intuitivo, atractivo y fácil de usar, manteniendo al mismo tiempo los elementos del juego original.

Aplicamos la metodología 4+1 para diseñar una arquitectura de software robusta y escalable. Esta metodología nos permitió abordar el proyecto desde múltiples perspectivas: lógica, desarrollo, procesos, física y escenarios, asegurando una cobertura completa de todos los aspectos del sistema.

El proyecto se divide en varios módulos (`buscaminas.py`, `casilla.py`, `main.py`, y `tablero.py`), cada uno enfocado en diferentes aspectos del juego, como la gestión del tablero, la lógica de las casillas y la interfaz de usuario. Los patrones de diseño y las buenas prácticas de desarrollo se aplican para mejorar la calidad del código.

Este proyecto no solo ofrecerá una experiencia de juego atractiva y desafiante sino que también se implementarán las buenas prácticas de desarrollo de software y diseño de arquitectura. Demostrando cómo la metodología y la planificación cuidadosa nos puede llevar a la creación de un software de alta calidad y fácil de mantener.

Diagramas

Los diagramas son la esencia de la visualización en el diseño de software, ofreciendo una narrativa clara de la arquitectura y las operaciones subyacentes de un sistema. En el desarrollo de nuestro juego de buscaminas, los diagramas no solo sirven para documentar la estructura del código y la interacción de sus componentes sino también para planificar y razonar sobre las modificaciones y mejoras futuras.

Esta sección del informe despliega una serie de diagramas críticos:

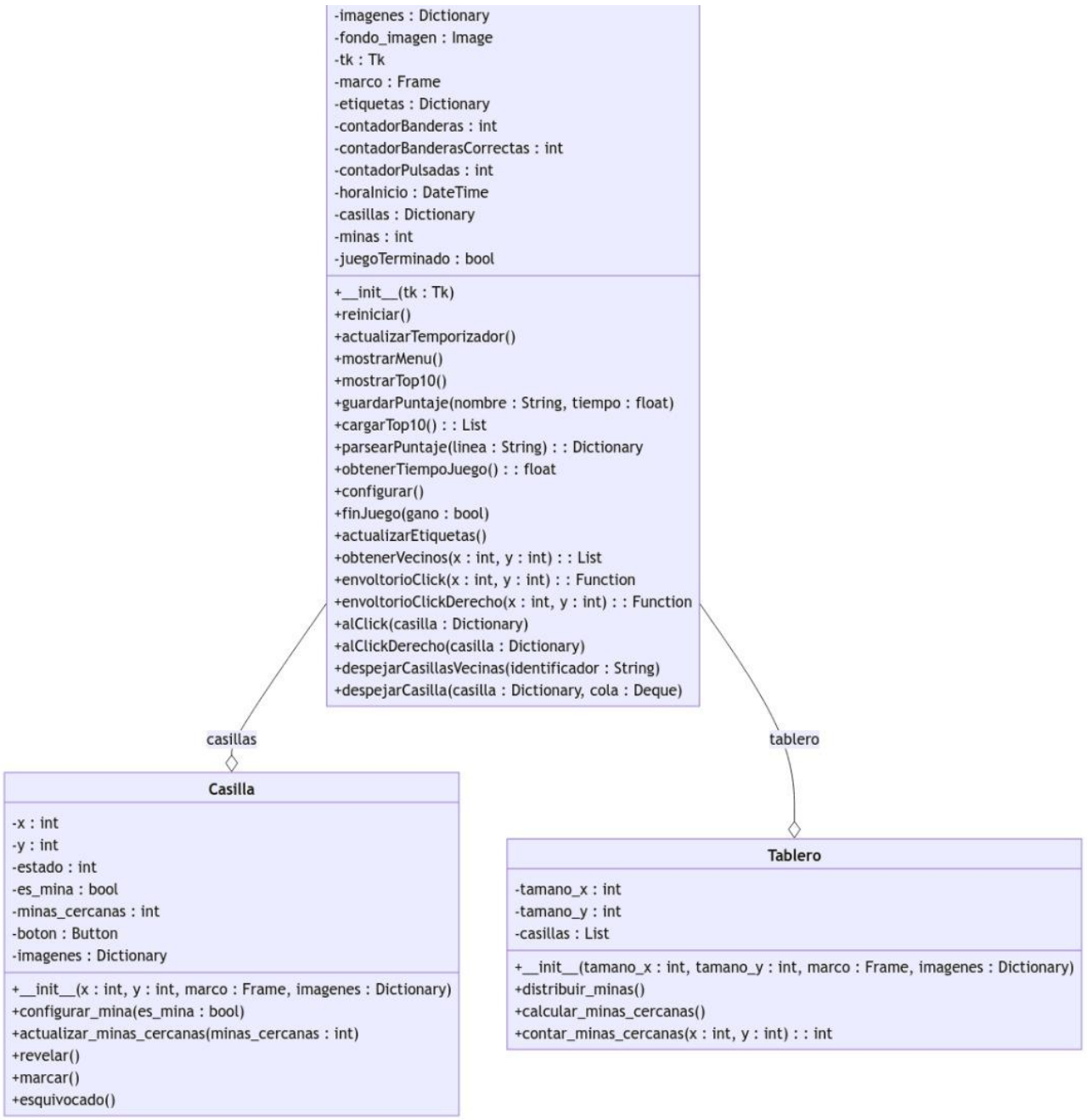
- Diagrama de Clases: Revela la organización orientada a objetos del juego, mostrando las clases, sus atributos y métodos, y cómo se interrelacionan.

- Diagrama de Secuencia: Ilustra el flujo de interacciones entre objetos a lo largo del tiempo, enfocándose en cómo los usuarios y el sistema interactúan durante una partida.

- Diagrama de Despliegue: Muestra cómo el software se distribuye sobre la infraestructura de hardware y se ejecuta en el entorno de producción.

- Diagrama de Componentes: Descompone el sistema en sus módulos lógicos, destacando la modularidad y la colaboración entre ellos.

Diagrama de Clases



El Diagrama de Clases para el juego de buscaminas ilustra cómo se organiza y estructura el software a nivel de clases, mostrando la relación entre los distintos componentes del juego y sus responsabilidades individuales.

Diagrama de Clases: Juego de Buscaminas

- Clase 'Juego':
    - Esta clase sirve como el controlador central del juego, donde se inicia y controla la partida.
    - Maneja los recursos gráficos, el estado del juego, y las interacciones del usuario a través de una serie de métodos que incluyen la inicialización, la actualización del temporizador, la gestión del menú y el procesamiento de las acciones del jugador.
- Clase 'Casilla':
    - Representa una casilla individual en el tablero de buscaminas, manteniendo su posición, estado y contenido.
    - Encapsula la lógica para la configuración y revelación de las casillas, así como la actualización del conteo de minas cercanas, fundamental para la mecánica del juego.
- Clase 'Tablero':
    - Organiza el conjunto de casillas y gestiona la lógica global del tablero, como la distribución de minas y el recuento de minas cercanas.
    - Actúa como un contenedor para las casillas y maneja las operaciones a nivel de tablero, tales como la inicialización y la determinación de la lógica de victoria o derrota.

La estructura del diagrama refleja un enfoque de diseño claro y modular, con clases bien definidas y responsabilidades específicas. La clase 'Juego' actúa como el núcleo que conecta la lógica de juego con la interfaz de usuario, mientras que 'Tablero' y 'Casilla' se ocupan de los detalles específicos del juego. Esta organización no solo promueve la claridad y la mantenibilidad del código sino que también facilita la expansión futura y la reutilización de componentes.

Diagrama de Secuencia

El Diagrama de Secuencia proporciona una visión del flujo de interacciones entre diferentes objetos a lo largo del tiempo en el juego de buscaminas, destacando la dinámica de la interacción del usuario y la respuesta del sistema. A continuación se detalla cada paso ilustrado en el diagrama para su inclusión en el informe:

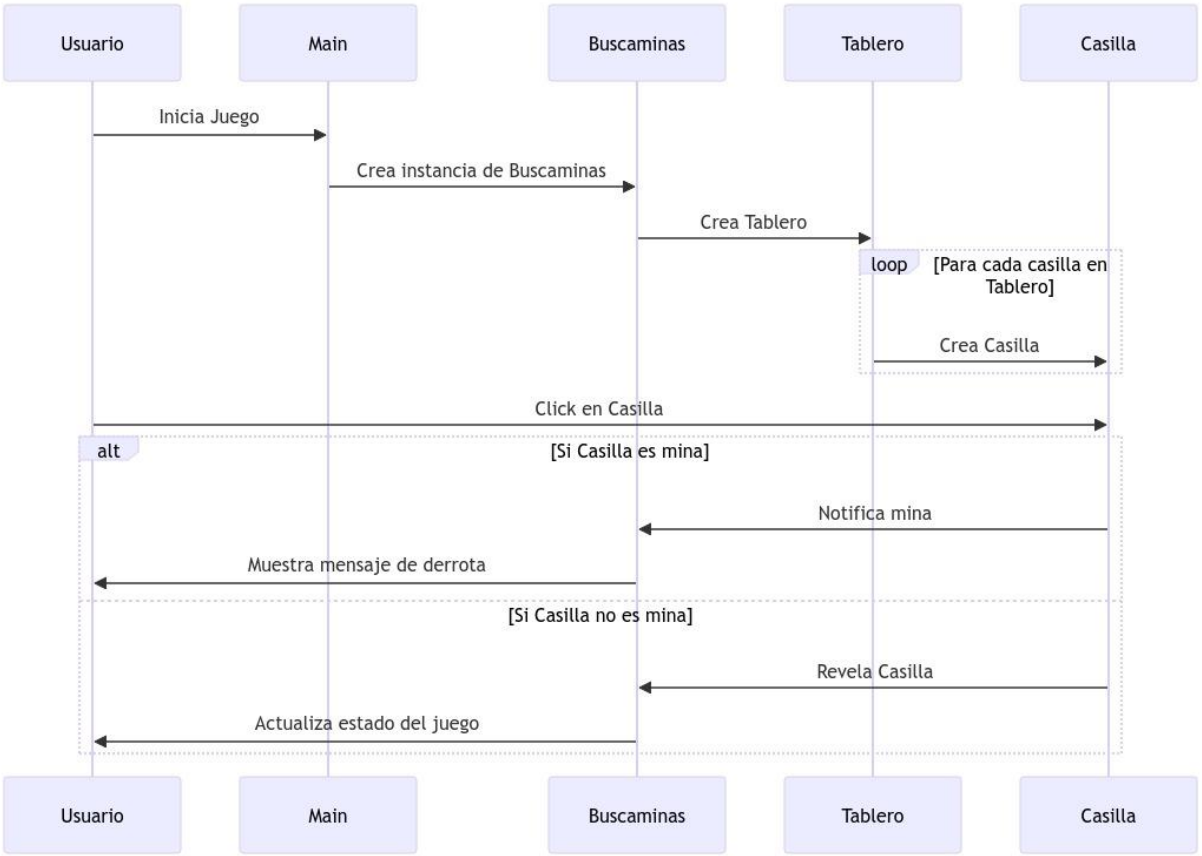
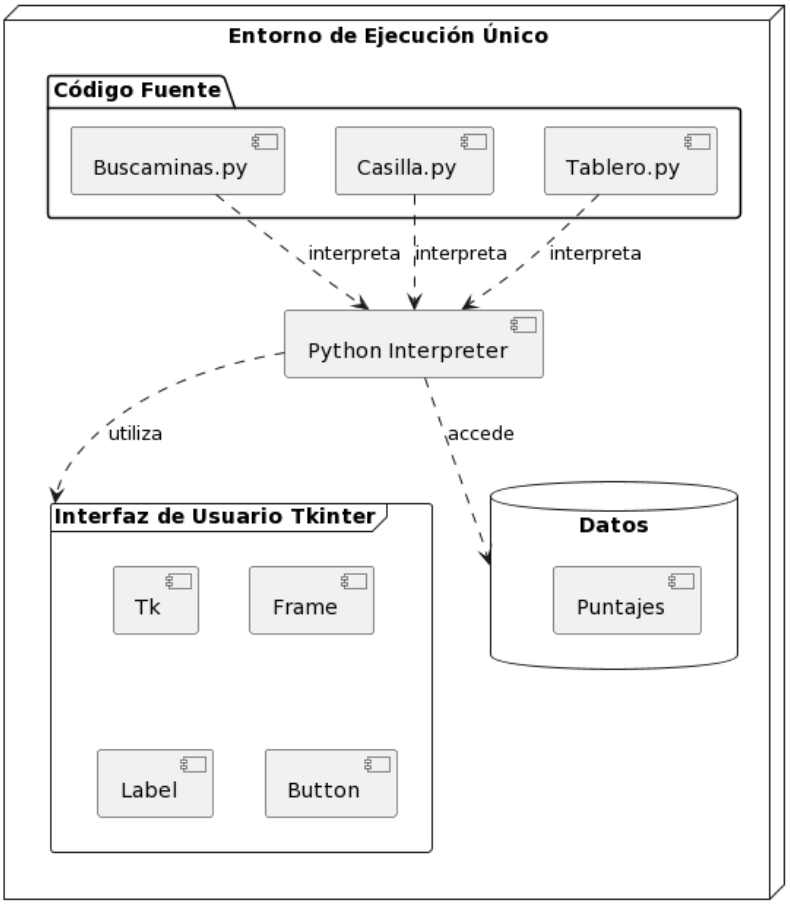


Diagrama de Secuencia: Interacción en Juego de Buscaminas

- Inicio del Juego:
    - El proceso comienza cuando el usuario inicia el juego.
    - La función principal (Main) responde creando una instancia de Buscaminas.
- Creación del Tablero:
    - Buscaminas procede a crear el Tablero, que es la estructura que sostiene el juego.
    - En un bucle, Tablero genera cada Casilla, estableciendo así el campo de juego.
- Interacción del Usuario:
    - El usuario interactúa con el sistema haciendo clic en una Casilla.
    - Dependiendo del estado de la Casilla, el sistema entra en una bifurcación (alt):
        - Si la Casilla es una mina:
            - El Tablero notifica la mina y el juego muestra un mensaje de derrota al usuario.
        - Si la Casilla no es una mina:
            - La Casilla se revela y el Tablero actualiza el estado del juego.

Este diagrama es crucial para comprender cómo se manejan las interacciones en tiempo real en el juego y es un testimonio de la implementación efectiva de la lógica de juego. Resalta la respuesta del sistema a las acciones del usuario y cómo estas acciones determinan el flujo de juego.

Diagrama de Despliegue



El Diagrama de Despliegue ilustra cómo se estructura y se ejecuta el juego de buscaminas en el entorno de ejecución, mostrando la relación entre el código fuente, el intérprete de Python, la interfaz de usuario y la persistencia de datos.

Diagrama de Despliegue: Juego de Buscaminas

En el entorno de ejecución único, tenemos tres archivos de código fuente fundamentales: Buscaminas.py, Casilla.py y Tablero.py. Cada uno de estos archivos representa un módulo distinto del juego:

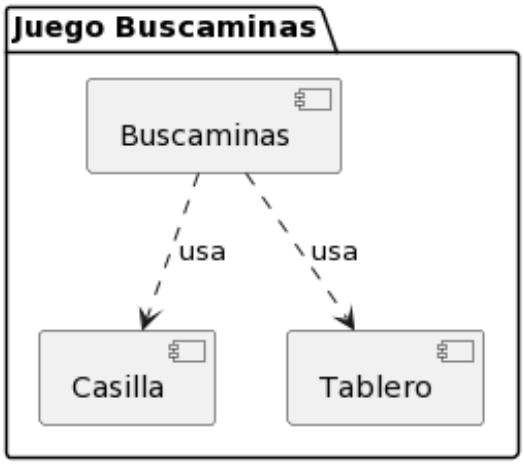
- Código Fuente:
    - Buscaminas.py contiene la lógica principal del juego.
    - Casilla.py gestiona el estado y comportamiento de cada casilla del tablero.
    - Tablero.py organiza las casillas y controla la lógica global del tablero.

Estos archivos son interpretados por el intérprete de Python, que ejecuta el código y gestiona la lógica de la aplicación.

- Interfaz de Usuario Tkinter:
    - La interfaz gráfica está construida con Tkinter, un toolkit de Python para la creación de interfaces gráficas.
    - Elementos como Frame, Label y Button son utilizados para construir la interfaz de usuario, permitiendo una interacción visual y receptiva con el jugador.
- Persistencia de Datos:
    - Los Datos se almacenan en una estructura que guarda Puntajes, manteniendo un registro de los resultados del juego.

El diagrama indica claramente cómo el código fuente se relaciona directamente con la interfaz de usuario y la base de datos de puntajes. Las flechas sólidas que conectan los archivos de código fuente con el intérprete de Python y luego con la interfaz de usuario de Tkinter representan la ejecución y utilización de los módulos del juego. Mientras que la línea punteada hacia el contenedor de Datos muestra el acceso a los puntajes, indicando una operación de lectura y/o escritura.

Diagrama de Componentes



El Diagrama de Componentes ofrece una perspectiva de alto nivel del juego de buscaminas, enfocándose en la organización y la interacción entre sus módulos principales. Este tipo de diagrama es crucial para entender cómo se ensamblan y se relacionan los componentes del software, proporcionando una vista del sistema desde la perspectiva de su construcción modular.

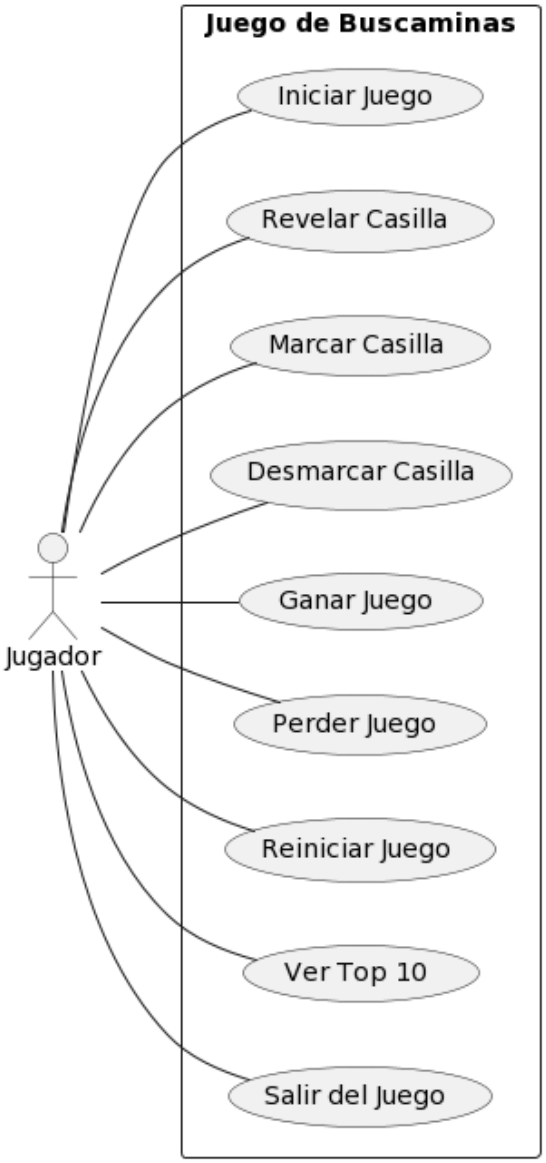
Diagrama de Componentes: Juego de Buscaminas

El diagrama muestra tres componentes principales:

- Componente 'Buscaminas':
  - Actúa como el componente principal del juego, encapsulando la lógica y las operaciones de alto nivel.
  - Se comunica con los componentes 'Casilla' y 'Tablero' para gestionar el flujo del juego.
- Componente 'Casilla':
  - Representa un módulo que gestiona la información y el comportamiento de una casilla individual dentro del tablero.
  - Interactúa con el componente 'Buscaminas' para actualizar el estado de cada casilla en respuesta a las acciones del usuario.
- Componente 'Tablero':
  - Es el componente que estructura la disposición de las casillas y controla la lógica del tablero.
  - Trabaja junto con el componente 'Buscaminas' para organizar el juego y manejar la lógica que afecta a todo el tablero.

Las líneas discontinuas entre 'Buscaminas' y los otros dos componentes indican una relación de uso, lo que significa que 'Buscaminas' utiliza los servicios y funcionalidades proporcionadas por 'Casilla' y 'Tablero'.

Diagrama de Casos de Uso



Representa las interacciones que el jugador puede tener con el juego de buscaminas. Cada caso de uso ilustra una función o una acción que el jugador puede realizar dentro del juego.

Este diagrama es fundamental para entender cómo el usuario interactúa con el juego y qué operaciones están disponibles. Es una representación gráfica de las funcionalidades ofrecidas al jugador, destacando la usabilidad y la accesibilidad del juego de buscaminas. La inclusión de este diagrama en el informe subraya el compromiso del equipo de desarrollo con la creación de una interfaz de usuario centrada en el jugador, que es intuitiva y fácil de navegar. Cada caso de uso describe una parte esencial de la experiencia del jugador, asegurando que todas las acciones posibles sean consideradas y bien definidas en el diseño del software.

Diagrama de Casos de Uso: Interacción del Jugador con el Juego de Buscaminas

- Iniciar Juego:
  - Representa la acción de comenzar una nueva partida.
- Revelar Casilla:
  - Corresponde a la acción de descubrir el contenido de una casilla específica en el tablero.
- Marcar Casilla:
  - Indica la función de marcar una casilla que el jugador sospecha contiene una mina.
- Desmarcar Casilla:
  - La acción de remover una marca previamente puesta sobre una casilla.
- Ganar Juego:
  - Este caso de uso se activa cuando el jugador ha revelado todas las casillas no minadas.
- Perder Juego:
  - Ocurre cuando el jugador revela una casilla que contiene una mina.
- Reiniciar Juego:
  - La función que permite al jugador comenzar una nueva partida desde cero.
- Ver Top 10:
  - Permite al jugador ver los diez mejores puntajes alcanzados en el juego.
- Salir del Juego:
  - La opción para cerrar el juego y salir de la aplicación.

Pattern Design

Los Pattern Design implementados en el código son:

- Modelo Vista Controlador (MVC)

Este popular patrón separa la lógica de negocio, los datos (Modelo), de la interfaz de usuario (Vista), utilizando controladores que gestionan la interacción y el flujo de datos entre ambos componentes, esto proporciona una mejor división del trabajo y una mejora de mantenimiento. En el Buscaminas se aplica separando claramente las reglas del juego, el tablero y casillas (modelo) de la interfaz gráfica en Tkinter (vista), con controladores como los manejadores de eventos de casillas que se encargan de conectar las acciones del usuario con operaciones que actualizan el estado del juego. Esta separación fomenta la modularidad, facilita el testing, y el mantenimiento del código.(Para algunos es considerado más un architectural pattern).

- Observer

Observer es un pattern design de comportamiento que te permite definir un mecanismo de suscripción para notificar a varios objetos sobre cualquier evento que le suceda al objeto que están observando, esto con el fin de poder promover la escalabilidad y modularidad. Se usa aquí para separar el modelo de datos (tablero, casillas) de la interfaz gráfica, permitiendo que cambios en los primeros se reflejen en la segunda de forma transparente.

- Singleton

Singleton es un patrón de diseño creacional que nos permite asegurarnos de que una clase tenga una única instancia, a la vez que proporciona un punto de acceso global a dicha instancia.Se implementa en la aplicación en la ventana principal Tkinter, de forma que se garantiza que habrá solo una ventana compartida por todas las demás partes del programa en lugar de crear múltiples ventanas innecesarias. Al asegurar un acceso único y controlado a la ventana raíz, se simplifica el código, se reduce el uso innecesario de memoria al no usar ventanas de más, y se tiene un control enfocado en cómo interactuar con la interfaz gráfica de usuario principal.

Conclusión

Al concluir el proyecto "Juego de Buscaminas", se destaca no solo por digitalizar un juego clásico sino también por ilustrar el impacto de un enfoque de desarrollo sistemático y prácticas óptimas. Con la utilización de Python y Tkinter se ha dado lugar a una interfaz de usuario accesible y atractiva, ofreciendo una experiencia de juego envolvente y retadora.

El despliegue de la metodología 4+1 en el diseño del software ha sido clave, ya que permitió una visión completa del sistema desde diferentes puntos de vista, resultando en un diseño lo suficientemente detallado, facilitando así una estructura modular para un fácil mantenimiento y futura expansión del software.

Integrando patrones de diseño y buenas prácticas, mejoraron la calidad del código y la eficiencia del sistema, beneficiando la experiencia del usuario y preparando el terreno para actualizaciones regulares.

Para las futuras mejoras, se sugiere una mayor separación y definición de las clases para mejorar la claridad del código y facilitar la extensión de funcionalidades. Además, la introducción de nuevas características como modos de juego variados y opciones multijugador podría aumentar significativamente el atractivo del juego.

En conclusión, este proyecto ha sido una experiencia que nos dio la posibilidad de adquirir conocimiento invaluable para demostrar cómo una planificación cuidadosa y el alineamiento a prácticas de arquitectura de software recomendadas pueden resultar en la creación de aplicaciones funcionales y atractivas. Este enfoque establece una base sólida para futuros proyectos de desarrollo de software.