|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UNIVERSIDAD AUTÓNOMA “TOMÁS FRÍAS” INGENIERÍA DE SISTEMAS** | | | | | |
| **ESTUDIANTE:** | Darwin Manuel Mamani Ojopi | C.I: | 10470829 | **PRÁCTICA N5 AUX** |
| **DOCENTE:** | Ing. Ditmar Castro Angulo | **MATERIA:** | SIS-211 G2 |
| **AUXILIAR:** | Univ. Gabriel Alejandro Garvizu Salas | **FECHA:** | 02/11/2024 |

1. Notificación de juego.

Las notificaciones en este juego son manejadas principalmente en el JLabel textLabel. El texto en esta etiqueta es actualizado para notificar al usuario del estado del juego, tal como: Tres en raya al inicio del juego, Turno de X o Turno de O para indicar de quien es el turno, X es el ganador! o O es el ganador! en el momento que el ganador es descubierto, e incluso Empate! si el estado finaliza en empate.

1. Bonificaciones de juego.

No aplica en este contexto porque el juego de “Tres en Raya” implementado no tiene bonificaciones ni recompensas. No hay un sistema de puntos, recompensas extras, o bonificaciones; como se vería en un juego sofisticado. Si prefieres añadir bonificaciones, deberías expandir al programa para agregar un sistema de puntos, logros o niveles.

1. Sistema de tareas.

Tareas es un término vago que se refiere a la realización de objetivos específicos o misiones en el contexto de un juego. Encima de este código, no hay un sistema de tareas explícito. Pero, piense en esto: ¿tú decir que el objetivo del jugador es conseguir tres en línea con la misma letra, ya sea “X” o “O”. Desde luego, estos serían tareas en el contexto de un juego. Para un sistema de tareas real, podrías añadir desafíos difíciles como, “gana en menos de 5 movimientos” o “logra ganar empleando solo una fila”.

1. Sistema de filtros de objetos de juego.

De hecho, el concepto de un filtro de objetos no se puede aplicar directamente a este código. No hay un set de objetos que se deba filtrar o seleccionar de alguna manera; sin embargo, en los juegos más complejos, un sistema de filtros se podría utilizar para filtrar los elementos según algún criterio específico. Lo más cercano al “filtro” que encontré en este código fue la lógica que selecciona qué celdas tienen un campo de texto vacío y, por lo tanto, deja que la piedra se mueva: if (tile.getText().equals("")) { tile.setText(currentPlayer); } Aquí, el “filtro” que aplicó este programa a los objetos es el hecho de que especifica las celdas previamente ocupadas al revisar si el texto de un botón comienza con una cadena vacía y, en ese caso, deja que un jugador ponga su piedra allí.