MASTERMIND

Descripción Diagrama de Classes

Projectes de Programació: Quadrimestre Primavera 2022-23

Grup 42.5:
Elsa Boix Solina
Joel Macías Rojas
Ricard Medina Amado
Marcel Sánchez Roca

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Descripción del diagrama de clases

1. Usuario

Breve descripción de la clase: Registra la información del usuario en nuestro sistema.

Cardinalidad: Una para cada usuario.

Descripción de los atributos:

- username – Nombre que identifica al usuario
- maxScore – Vector que guarda el orden de las puntuaciones de las partidas realizadas por el usuario.
- partidas – Vector que guarda las fechas de les partidas jugadas por el usuario.

Descripción de las relaciones:

- Relación de asociación con la clase “Partida”: Indica quién es el que realiza la partida.
- Relación de asociación con la clase “CtrlUsuario”: Indica que el usuario es gestionado por el CtrlUsuario.

2. CtrlUsuario

Breve descripción de la clase: Tiene el conjunto de instancias de de “Usuario”

Cardinalidad: Una

Descripción de los atributos:

- usuarios – ArrayList con los nombres de los usuarios del sistema.
- userAct – Instancia del usuario que interactúa actualmente con el sistema.

Descripción de las relaciones:

- Relación de asociación con la clase “CtrlDominio”: El Controlador Domini crea una instancia de Controlador Usuario y se utiliza en las funcionalidades necesarias del sistema.
- Relación de asociación con la clase “Usuario”: Gestiona a los usuarios del sistema.

- Relación de asociación con la clase “CtrlPartida”: Necesario para que el usuario pueda interactuar con la partida.

3. Partida

Breve descripción de la clase: Registra la información de una partida que ha sido creada por la aplicación.
Cardinalidad: Una por cada partida.
Descripción de los atributos:

- data – Date que sirve com identificador de la partida.
- puntos – Int que indica la puntuación resultante de la partida.
- ayuda – Bool que si es falso indica que no se quiere ayuda en la partida, true si se quiere ayuda en la partida.
- estadoPartida – Indica el estado actual de la partida, RUNNING, PAUSED, SAVED.
- solutions – ArrayList que contiene la solución de la partida.
- nivel – Indica el nivel de dificultat de la partida: BAJO, MEDIO, ALTO
- username – Indica el nombre del usuario encargado de la partida.
- turnos – Indica el turno actual de la partida.

Descripción de las relaciones:

- Relación de asociación con la clase “Usuario”: Indica de quién es la partida.
- Relación de asociación con la clase “CtrlPartida”: Indica que la partida es gestionada por el controlador de partida.
- Relación de asociación con la clase “Ranking” Indica que la partida estará en un ranking.
- Relación de asociación con la clase “NivelDificultad”: Necesaria para tratar la partida de una manera u otra dependiendo del nivel de dificultat.
- Relación de asociación con la clase “Turno”: Necesaria para tratar de una manera u otra dependiendo de si ha de jugar como CodeBraker o CodeMaker.
- Relación de asociación con la clase “Combinación”: Necesaria para tener la funcionalidad de enviar combinaciones i soluciones.

Nombre de la clase: CtrlPartida
Breve descripción de la clase: Tiene el conjunto de instancias de la clase Partida
Cardinalidad: Una
Descripción de los atributos:

- partidaActual – Instancia actual de la partida que se está tratando en el sistema.

Descripción de las relaciones:

- Relación de asociación con la clase “Partida”. Gestiona las partidas del sistema
- Relación de asociación con la clase “CtrlUsuario”. Necesario para que el usuario pueda interactuar con la partida.

4. CtrlDomini

Breve descripción de la clase: Se utiliza para la funcionalidad de todas las clases y poder probar el funcionamiento de las mismas.
Cardinalidad: Una
Descripción de los atributos:
Descripción de las relaciones:

- Relación de asociación con “CtrlUsuario”: el controlador de dominio crea una instancia de CtrlUsuario y accede a ella cuando quiere empezar con las funcionalidades del sistema.
- Relació d’associació amb “Ranking”: el controlador de dominio crea y gestiona el ranking del sistema.

5. Turno

Breve descripción de la clase: Se utiliza para dar unas u otras funcionalidades dependiendo de si se es CodeMaker o CodeBreaker.
Cardinalidad: Dos por Partida
Descripción de los atributos:

- rol: bool que false=CodeBreaker, true=CodeMaker
- combinations: ArrayList con las combinaciones que se han enviado en la partida-

Descripción de las relaciones:

- Relación de asociación con “Partida”: Necesaria para tener todas las funcionalidades relacionadas con enviar combinaciones.
- Relación de asociación con “Combinacion”: Necesaria para poder crear combinaciones.

6. Combinacion

Breve descripción de la clase: Se utiliza para crear las combinaciones que servirán para la funcionalidad del juego

Cardinalidad: Entre 4-12 por partida. (Mìnimo 2 de CodeBreaker y 2 de CodeMaker)

Descripción de los atributos:

- combination: ArrayList de colores que crean la combinación.

Descripción de las relaciones:

- Relación de asociación con “Turno”: Necesaria para que Turno pueda enviar combinaciones a Partida.
- Relación de asociación con “Partida”: Necesaria para que Partida tenga una solución.

7. Ranking

Breve descripción de la clase: Se utiliza para tener un registro de las mejores puntuaciones de las partidas del sistema.

Cardinalidad: 3, Uno por cada nivel de dificultad.

Descripción de los atributos:

- posiciones: map que se encarga de ordenar las puntuaciones de cada partida y las asocia al usuario que ha jugado la partida..

Descripción de las relaciones:

- Relación de asociación con CtrlDomini; Necesaria para la gestión del mismo.

- Relación de asociación con partida: Necesaria para poder obtener tanto la puntuación de la partida como el usuario responsable de ella.

8. NivelDificultad

Breve descripción de la clase: Contiene los métodos comunes para los diferentes tipos de dificultad, además de las funciones encargadas de las comprobaciones entre envíos y soluciones.

Cardinalidad:

- Una por cada partida

Descripción de los atributos:

- numColumnas – Número de columnas que tendrá el juego.
- numColores – Número de colores que se podrá elegir para jugar.
- sePuedeRepetir – Bool que si true == se pueden repetir colores tanto en la solución como en la combinación, de lo contrario == false.
- turn – Indica el turno en que esta la partida.
- solución – solución de la partida

Descripción de las relaciones:

- Relación de asociación con “Partida”: Necesaria para que la Partida pueda disponer de una lógica para poder jugar e interactuar con la máquina.
- Relación de asociación con “Maquina”: Necesaria para poder acceder al algoritmo encargado de hallar la solución cuando se juega como *codemaker.*

9. NivelDificultadBajo

Breve descripción de la clase: Contiene los métodos específicos del nivel de dificultad bajo, dónde no hay repeticiones y el número total de columnas son 4.

Cardinalidad:

- 0..Una por cada partida

Descripción de los atributos:

- numColumnas – Número de columnas que tendrá el juego.
- numColores – Número de colores que se podrá elegir para jugar.

- sePuedeRepetir – Bool que si true == se pueden repetir colores tanto en la solución como en la combinación, de lo contrario == false.
- turn – Indica el turno en que esta la partida.
- solucion – solución de la partida

Descripción de las relaciones:

Recibe la herencia de la clase padre “NivelDificultad”: Necesaria para evitar repetición de código y darle atributos propios de la clase como número de columnas o nivel de dificultad que se requerirán en 5Guess para así adaptarse a las especificaciones del nivel de dificultad.

10. NivelDificultadMedia

Breve descripción de la clase: Contiene los métodos específicos del nivel de dificultad medio, dónde sí hay repeticiones y el número total de columnas son 4.

Cardinalidad:

- 0..Una por cada partida

Descripción de los atributos:

- numColumnas – Número de columnas que tendrá el juego.
- numColores – Número de colores que se podrá elegir para jugar.
- sePuedeRepetir – Bool que si true == se pueden repetir colores tanto en la solución como en la combinación, de lo contrario == false.
- turn – Indica el turno en que esta la partida.
- solucion – solución de la partida

Descripción de las relaciones:

- Recibe la herencia de la clase padre “NivelDificultad”: Necesaria para evitar repetición de código y darle atributos propios de la clase como número de columnas o nivel de dificultad que se requerirán en 5Guess para así adaptarse a las especificaciones del nivel de dificultad.

11. NivelDificultadAlta

Breve descripción de la clase: Contiene los métodos específicos del nivel de dificultad dificultad alto, dónde sí hay repeticiones y el número total de columnas son 5.

Cardinalidad:

- 0..Una por cada partida

Descripción de los atributos:

- numColumnas – Número de columnas que tendrá el juego.
- numColores – Número de colores que se podrá elegir para jugar.
- sePuedeRepetir – Bool que si true == se pueden repetir colores tanto en la solución como en la combinación, de lo contrario == false.
- turn – Indica el turno en que esta la partida.
- solucion – solución de la partida
- envioActual –

Descripción de las relaciones:

- Recibe la herencia de la clase padre “NivelDificultad”: Necesaria para evitar repetición de código y darle atributos propios de la clase como número de columnas o nivel de dificultad que se requerirán en 5Guess para así adaptarse a las especificaciones del nivel de dificultad.

12. FiveGuess

Breve descripción de la clase: Contiene los métodos para, mediante el algoritmo de *Five Guess,* generar los envíos cuando el jugador crea una solución como *codemaker.*

Cardinalidad:

- Una por cada partida

Descripción de los atributos:

- possibleCodes – Contiene las diferentes combinaciones de números que podrían ser la solución.
- totalcombinacionesPosibles – Contiene todas las diferentes combinaciones que se pueden generar en el nivel de dificultad especificado.

- solucionesEnviadas – Contiene los envíos realizados por el algoritmo para encontrar la solucion
- enviosCandidatos – Se genera en cada ronda, contiene las combinaciones más adecuadas para ser enviadas.
- turn - turno actual
- envioActual - Última combinación comprobada,
- solucion - Solución que el algoritmo debe encontrar
- nivel - Nivel de dificultad, 1 = Bajo, 2 = Medio, 3= Alto.
-

Descripción de las relaciones:

- Relación de asociación con Máquina: Necesaria para una vez implementado el otro algoritmo, facilitar las comunicaciones entre

13. Juego

Breve descripción de la clase: Se encarga de mostrar las reglas del juego y el criterio que se usa en el sistema de puntuación.
Cardinalidad: Una
Descripción de los atributos:

- InformaciónPuntuación: Texto que explica el criterio de puntuación.
- IndormaciónSistema: Texto que explica como funciona el juego.

Descripción de las relaciones:

- Relación de asociación con CtrlDomini: Necesaria para poder proporcionar las informaciones de las reglas del juego.

14. HistorialPartidas

Breve descripción de la clase: Se encarga de tener un historial con todas las partidas, independientemente del estado de la partida.
Cardinalidad: Una
Descripción de los atributos:

- partidas: ArrayList que contiene el usuario asociado a la partida y la fecha que identifica la partida.

Descripción de las relaciones:

- Relación de asociación con CtrlDomini: Necesaria para la gestión de la clase.
- Relación de asociación con CtrlPartida: Necesaria para obtener la fecha de la partida.

14. HistorialPartidasGuardadas

Breve descripción de la clase: Se encarga de tener un historial con todas las partidas guardadas en el sistema en ese momento.

Cardinalidad: Una

Descripción de los atributos:

- partidas: ArrayList que contiene el usuario asociado a la partida y la fecha que identifica la partida.

Descripción de las relaciones:

- Relación de asociación con CtrlDomini: Necesaria para la gestión de la clase.
- Relación de asociación con CtrlPartida: Necesaria para obtener la fecha de la partida.