Índice

1. Descripción del diagrama de clases

1.1. Clases de dominio

1.1.1. ControladorDominio

- Nombre de la classe:
  - ControladorDominio
- Breve descripció de la classe:
  - Controlador global de la capa de dominio encargado de comunicar esta capa con las demás mediante sus respectivos controladores permitiendo la arquitectura de tres capas (Presentación,Dominio,Persistencia).
- Cardinalidad:
  - Singleton, uno único.
- Descripción de los atributos:
  - ctrl (ControladorDominio): Instancia singleton.
  - controladorPersistencia (ControladorPersistencia): Instancia del controlador de persistencia.
  - rankingGlobal (RankingGlobal): Instancia del RankingGlobal.
  - recordsMovimientos (RecordsMovimientos): Instancia de RecordsMovimientos.
  - partida (IPartida): Instancia de la partida de tipo IPartida.
  - usuarioActivo (Perfil): Objeto de tipo Perfil del usuario activo.
  - Usuarios (HashMap<String,Perfil): Lista de usuarios del sistema.
  - partidaActiva (Partida): Objeto de tipo Partida de la partida activa.
  - idUltimaPartida (Integer): Id de la ultima partida creada.
- Descripción de las relaciones:
  - Relación de asociación con la clase “Perfil”: Indica el perfil del usuario activo.
  - Relación de asociación con la clase “IPartida”: Permite la comunicación mediante la interfaz de Partida con la partida actual.
  - Relación de asociación con la clase “RecordsMovimientos”: Indica la instancia de RecordsMovimientos del sistema.
  - Relación de asociación con la clase “RankingGlobal”:

Indica la instancia de RankingGlobal del sistema.
    - Relación de asociación con la clase “Partida”: Indica la partida cargada actualmente.
- Descripción de los métodos:
    - creaPartida(List<Object>):: int: Método encargado de ordenar la creación de una partida de acuerdo a la configuración especificada como parámetro.
    - cargaPartida(Integer):: void: Método encargado de cargar la partida con id especificado como parámetro.
    - ejecutarTurno(List<Integer>):: int: Método ejecutor de turnos que permite recibir como parámetro un código que se tratara de forma correcta dependiendo de la situación y configuración de la partida activa.
    - obtenPuntuacion():: double: Método encargado de obtener la puntuación de la partida activa.
    - iniciarSesion(String,String):: int: Método encargado de realizar el inicio de sesión a partir del nombre de usuario y la contraseña especificados en el primer y segundo parámetro respectivamente.
    - registrar(String,String):: int: Método encargado de realizar el registro de un perfil mediante el nombre de usuario y la contraseña especificados en el primer y segundo parámetro respectivamente.
    - cerrarSesion(String,String):: void: Método encargado de cerrar la sesion cargada/iniciada.
    - inicializarCtrlDominio():: void: Método encargado de inicializar el controlador de dominio estableciendo las variables iniciales y realizando la carga de datos guardados en persistencia para restablecer el estado de la anterior ejecución.

1.1.2. IPartida y FactoriaPartida

- Nombre de la classe:
    - Partida y FactoriaPartida
- Breve descripció de la classe:
    - Clases responsables de desacoplar las variantes de partida del resto del sistema mediante el patrón Simple Factory. IPartida es una interfaz implementada por PartidaCodemaker y PartidaCodebreaker, en cambio, FactoriaPartida es un singleton.
- Cardinalidad:
    - IPartida: Uno para cad partida.
    - FactoriaPartida: Singleton, uno único.
- Descripción de los atributos:
        - *IPartida sin atributos*

        - *FactoriaPartida (instancia Singleton e instancia IPartida)*
- Descripción de las relaciones:
    - Relación de implementacion con las variantes de Partida “PartidaCodemaker/PartidaCodebreaker”: Indica que las variantes de “Partida” implementan esta interficie.
- Descripción de los métodos:
    - (FactoriaPartida) creaPartida(int, List<Object>, Perfil):: IPartida: Método creador de partidas que dependiendo de la configuración establecido en el segundo parámetro, se crea una PartidaCodemaker o Codebreaker con el identificador específico del primer parámetro y se asocia al perfil indica en en el tercer parámetro.
    - (IPartida) ejecutarTurno(List<Integer>):: Integer: Método abstracto implementado por las variantes de Partida encargado de ejecutar el turno.
    - (IPartida) calculaPuntuacion():: Integer: Método abstracto implementado por las variantes de Partida responsable de calcular y devolver la puntuación de la partida en cuestión.

1.1.3. Partida

- Nombre de la classe:
    - Partida
- Breve descripció de la classe:
    - Clase central del diagrama, creada y configurada por el usuario, y contenedora de su tablero del juego.
- Cardinalidad:
    - Una para cada partida creada por el usuario.
- Descripción de los atributos:
    - id: Identificador de la partida.
    - acabada: Booleano indicador del estado de la partida (finalizada : en curso).
    - codigoSecreto: Lista de colores correspondientes al codigo secreto de la partida.
    - puntuacion: Variable almacenadora de la puntuación de la partida.
    - configuracion: Objeto de tipo “Configuracion” correspondiente a la configuración de la partida.
    - tablero: Objeto de tipo “Tablero” correspondiente al tablero de la partida.
    - perfil: Objeto de tipo “Perfil” correspondiente al perfil de la partida.
    - pistasDisponibles: Entero que contiene el numero de pistas disponibles:

    - ganada: Booleano que contiene el estado de victoria de la partida.
- Descripción de las relaciones:
    - Relación de asociación con la clase “Configuración”: Indica la configuración aplicada a la partida.
    - Relación de asociación con la clase “Tablero”: Indica qué tablero contiene la información de la partida, correspondiente al tablero físico.
    - Relacion de asociación con la clase “Perfil”: Indica el perfil asociado a la partida.
- Descripción de los métodos:
    - Partida(int,List<Object, Perfil):: Partida: Constructor de partida que precisa de tres parámetros: (Identificador de partida (“id”) : Lista contenedora de los atributos de configuración de la partida y el perfil asociado a la partida)
    - generaResultado(List<Integer>):: List<Integer>: Método para obtener el resultado correspondiente a la adivinanza (parámetro List<Integer>) comparada con el código secreto de la partida.

- compruebaValidezFila(List<Integer>):: void: Método que comprueba la validez de la longitud de la fila introducida respecto a la configuración de la partida.
- avanzarTurno():: void: Método que incrementa en uno el turno actual (variable de “Tablero”), mediante su función incrementado (función de “Tablero”).
- calculaPuntuacion():: double: Método abstracto encargado de calcular la puntuacion de partida dependiendo de la variante de esta.
- ejecutarTurno(List<Integer>):: int: Método abstracto encargado de ejecutar un turno y devolver el estado de la partida como int
  - {0: en curso, -1, perdida, 1 ganada), dependiendo de la variante de partida.
- obtenPista():: List<Integer>: Método abstracto encargado de obtener la pista correspondiente a la variante de partida.

1.1.4. PartidaCodemaker

- Nombre de la classe:
  - PartidaCodemaker
- Breve descripció de la classe:
  - Clase que hereda de la clase “Partida”, y que representa la variación de partida correspondiente al rol de Codemaker.
- Cardinalidad:
  - Una para cada partida creada por el usuario en que este juegue como Codemaker.
- Descripción de los atributos:
  - maquina (Maquina): Objeto de tipo “Maquina” con el que se ejecutarán los algorismos implementados (FiveGuess : Genetics).
- Descripción de las relaciones:
  - Relación de generalización con la clase “Partida”: Indica que es una variante de partida.
  - Relación de asociación con la clase “Maquina”: Indica que por cada PartidaCodemaker, se le atribuye una Maquina para escoger el algoritmo y configuración de este a ejecutar.
  - Relación de implementación con la clase “IPartida”: Indica que implementa la interfaz IPartida y por tanto sus métodos definidos.
- Descripción de los métodos:
  - PartidaCodemaker(int,List<Object, Perfil):: PartidaCodemaker: Constructor de partida que precisa de tres parámetros: (Identificador de partida (“id”) : Lista contenedora de los

        - atributos de configuración de la partida), con el añadido de configurar (configuraMaquina() la “Maquina” perteneciente a esta partidaCodemaker y por último el perfil asociado a la partida.
    - configuraMaquinaGenetic/FiveGuess():: void: Métodos que configura la máquina de acuerdo a la configuración de la partida.
    - calculaPuntuacion():: Integer: Método que calcula y devuelve la puntuación de la partida correspondiente al usuario como Codemaker.
    - ejecutarTurno(List<Integer>):: Integer: Método que realiza la ejecución del turno del tipo partidaCodemaker, requiriendo en caso de justo haber iniciado la partida, el código secreto como parámetro, de lo contrario, se lanza una excepción. Seguidamente, obtiene la adivinanza realizada por la máquina, al igual que su resultado (obtenResultado()), ahora bien, este resultado debe ser informado por el usuario; informa ambos al tablero, avanza el turno (avanzarTurno()) y comprueba si la partida debe acabar, desembocando en victoria, derrota o continuación.
    - sincronizarConfiguracionGenetic():: void: Método encargado de sincronizar la configuración de la partida con la de la maquina.
    - obtenPista():: List<Integer>: Método encargado de obtener la pista correspondiente al rol codemaker.

1.1.5. PartidaCodebreaker

- Nombre de la classe:
    - PartidaCodebreaker
- Breve descripció de la classe:
    - Clase que hereda de la clase “Partida”, y que representa la variación de partida correspondiente al rol de Codebreaker.
- Cardinalidad:
    - Una para cada partida creada por el usuario en que este juegue como Codebreaker.
- Descripción de los atributos:
    - codigoRevelador (Integer[]) : Array de integers que contiene el codigo revelado hasta el momento.
    - posicionesReveladas (Integer[]): Array de integers que contienen que posiciones se han revelado.
    - numeroPosicionesReveladas (Integer): Entero que contiene el numero de posiciones reveladas hasta el momento.
- Descripción de las relaciones:

    - Relación de generalización con la clase “Partida”: Indica que es una variante de partida.
    - Relación de implementación con la clase “IPartida”: Indica que implementa la interfaz IPartida y por tanto sus métodos definidos.
- Descripción de los métodos:
    - PartidaCodebreaker(int,List<Object, Perfil):: PartidaCodebreaker: Constructor de partida que precisa de tres parámetros: (Identificador de partida (“id”) : Lista contenedora de los atributos de configuración de la partida y el perfil asociado a la partida), con el añadido de generar el código secreto (generarCodigoSecreto()) e informando a la “Partida”.
    - calculaPuntuacion():: Integer: Método que calcula y devuelve la puntuación de la partida correspondiente al usuario como Codebreaker.
    - ejecutarTurno(List<Integer>):: Integer: Método que realiza la ejecución del turno del tipo partidaCodebreaker, requiriendo en cada turno, la adivinanza como parámetro, de lo contrario, se lanza una excepción. Seguidamente, se obtiene el resultado (obtenResultado()), informa ambos al tablero, avanza el turno (avanzarTurno()) y comprueba si la partida debe acabar, desembocando en victoria, derrota o continuación.
    - generarCodigoSecreto(int, int):: List<Integer>: Método que genera y devuelve un código al azar correspondiente a la configuración de la partida necesaria (int: longitud del código, int: número de colores diferentes).
    - obtenPista():: List<Integer>: Método encargado de obtener la pista correspondiente al rol de codebreaker.

1.1.6. Tablero

- Nombre de la classe:
    - Tablero
- Breve descripció de la classe:
    - Clase contenedora del tablero de adivinanzas y resultados, que representa el tablero físico.
- Cardinalidad:
    - Uno para cada partida creada por el usuario.

- Descripción de los atributos:
  - numeroTurnoActual: Número de turno actual de la partida jugada en ese tablero.
  - tablero: Matriz correspondiente al tablero de adivinanzas.
  - tableroResultados: Matriz correspondiente al tablero donde se informan los resultados relativos a las adivinanzas.
- Descripción de las relaciones:
  - Relación de asociación con la clase “Partida”: Permite que cada “Partida” posea su tablero.
- Descripción de los métodos:
  - Tablero(int, int): Constructor de tablero e inicializador, tanto del tablero de adivinanzas como el de resultados.
  - informaAdivinanza(List<Integer>): Informa en “tablero”, en la posición del turno actual, la adivinanza pasada como parámetro.
  - informaResultado(List<Integer>): Informa en “tableroResultados”, en la posición del turno actual, el resultado pasado como parámetro.
  - avanzarTurno(): Incrementa en uno el atributo numeroTurnoActual.
  - obtenAdivinanza(List<Integer>): Obtiene la adivinanza de “tablero”, en la posición del turno actual.
  - obtenResultado(List<Integer>): Obtiene el resultado de “tableroResultados”, en la posición del turno actual.

1.1.7. Configuración

- Nombre de la classe:
  - Configuración
- Breve descripció de la classe
  - Clase contenedora de la configuración de las partidas.
- Cardinalidad:
  - Una para cada partida creada por el usuario.
- Descripción de los atributos:

    - numeroTurnos (Integer): Número de turnos máximos de la partida.
    - dificultad (Integer): Nivel de dificultad de la partida.
    - nivelAyuda (Integer): Nivel de ayuda al usuario en la partida.
    - numeroColores (Integer): Número de colores diferentes posibles en la partida.
    - longitudCodigo (Integer): Longitud de los codigos de la partida.
    - esCodemaker (Boolean): Indica si el jugador es Codemaker o de lo contrario, Codebreaker.
- Descripción de las relaciones:
    - Relación de asociación con la clase “Partida”: Permite que cada “Partida” posea su configuración.
- Descripción de los métodos:
    - Configuracion(List<Object>):: Configuración: Constructor mediante un parámetro lista que contiene los valores de la configuración

1.1.8. Máquina y derivadas

- Nombre de las clases:
    - Máquina, MáquinaGenetic y MáquinaFiveGuess
- Breve descripció de la classe:
    - Clases contenedoras de los algoritmos Genetic y FiveGuess, junto con la interfaz de estas “Máquina”.
- Cardinalidad:
    - Uno por cada “PartidaCodemaker”.
- Descripción de los atributos:
        - *– sin atributos –*
- Descripción de las relaciones:
    - Relación de Máquina de asociación con la clase “PartidaCodemaker”: Indica que cada partida en la cual el jugador interactúe como Codemaker “PartidaCodemaker” tiene su máquina de algoritmos asignada.
- Descripción de los métodos:
    - solve(List<Integer>):: List<List<Integer>>: Método definido por la interfaz e implementado por las máquinas.
    - **Genetic**: Tomando el código secreto como parámetro en forma de lista de enteros, se encarga de la ejecución del algoritmo escogido, y devuelve una lista de códigos correspondientes a la lista de mejores códigos, uno por generación, conteniendo un máximo de maxSteps códigos.
    - **FiveGuess**: Tomando el código secreto como parámetro en forma de lista de enteros, se encarga de la ejecución del algoritmo escogido, y devuelve una lista de los códigos que la maquina haya usado para descifrar el código o en caso de no haber sido capaz los que haya usado para intentarlo.

- MáquinaGenetic(int, int, int):: MáquinaGenetic: Constructor de MáquinaGenetic en cuyos parámetros corresponden al número de colores diferentes posibles, a los maxSteps o número de generaciones máximas y a la longitud de los códigos, correspondiente a la partida donde se ejecuta el algoritmo.
- MáquinaFiveGuess(int, int, int):: MáquinaFiveGuess: Constructor de MaquinaFiveGuess en cuyos parámetros corresponden al número de colores diferentes posibles, a los turnos maximos y la longitud del codigo.

1.1.9. Color

- Nombre de la classe:
  - Color
- Breve descripció de la classe:
  - Clase representativa de los colores que contienen los códigos (usada en el entorno del algoritmo Genetic).
- Cardinalidad:
  - Tantos como la longitud de los códigos tratados en el algoritmo, para cada código.
- Descripción de los atributos:
  - valor (Object): Atributo en forma Object, que contiene el número/valor del color en cuestión.
  - valorMaximoColor(int): Atributo correspondiente al número máximo de colores diferentes que puede contener un código.
- Descripción de las relaciones:
  - Relación de asociación con la clase “Código”: Permite dar forma a la relación que hace de un color parte de un código compuesto por múltiples colores.
- Descripción de los métodos:
  - colorRandom():: Integer: Método que devuelve un entero al azar, generado entre [0,maxColorValue).

1.1.10. Código

- Nombre de la classe:
  - Código
- Breve descripció de la classe:
  - Clase representativa de los códigos que contienen las poblaciones de códigos (usada en el entorno del algoritmo Genetic).
- Cardinalidad:

    - Tantos como la longitud de las poblaciones de códigos, más auxiliares usados.
- Descripción de los atributos:
    - longitudCodigo (Integer): Atributo en forma Integer, que contiene la longitud que debe tener el código.
    - fitness (Double): Atributo que favorece el rendimiento del algoritmo, y contenedor del fitness del código (parámetro del algoritmo Genetic).
    - colores (Vector<Color>): Vector de colores que contiene los colores que forman el código.
- Descripción de las relaciones:
    - Relación de asociación con la clase “PoblaciónCodigos”: Permite dar forma a la relación que hace de un código parte de una población de códigos.
    - Relación de asociación con la clase “CalculoFitness”: Indica que la clase CalculoFitness contiene un código perteneciente, en este caso, al código secreto.
- Descripción de los métodos:
    - Código():: Código: Constructor de Código que además inicializa el vector de colores.
    - informaColorRandom(int):: void: Método encargado de insertar un color generado al azar en la posición, indicada por el parámetro, en el vector de colores.
    - inicializaCódigo():: void: Método responsable de inicializar el código con valores al azar.
    - fitnessCódigo():: double: Método responsable de calcular en caso que este valga 0, el atributo fitness del código

1.1.11. PoblaciónCodigos

- Nombre de la classe:
    - PoblaciónCodigos
- Breve descripció de la classe:
    - Clase representativa de las poblaciones de códigos (usada en el entorno del algoritmo Genetic).
- Cardinalidad:
    - Tantos como generaciones y/o steps evolucione una población, además de auxiliares.

- Descripción de los atributos:
  - códigos (Código[]): Array de objetos de tipo Código, que contiene los códigos de la población.
- Descripción de las relaciones:
  - Relación de asociación con la clase “Código”: Permite dar forma a la relación que hace de una población de códigos el conjunto de códigos.
- Descripción de los métodos:
  - PoblaciónCodigos(int, boolean):: PoblaciónCodigos: Constructor de PoblaciónCodigos que atribuye el tamaño de la población al primer parámetro y la decisión de inicializarla como segundo.
  - mejorCodigo ():: Código: Método encargado de devolver el código más fit/apto de la población en cuestión.

1.1.12. CalculoFitness

- Nombre de la classe:
  - CalculoFitness
- Breve descripció de la classe
  - Clase responsable de llevar a cabo el calculo del fitness/aptitud de los códigos respecto el código secreto (usada en el entorno del algoritmo Genetic).
- Cardinalidad:
  - Una única instància, clase estática.
- Descripción de los atributos:
  - codigoSecretoGenetic (Código): Atributo de tipo Código que contiene el código secreto y/o solución con la que comparar los códigos recibidos.
- Descripción de las relaciones:
  - Relación de asociación con la clase “Código”: Indica que la clase CalculoFitness contiene un código perteneciente, en este caso, al código secreto.
- Descripción de los métodos:
  - maxFitness():: double: Método que devuelve el valor máximo posible de fitness obtenible por los códigos.
  - fitnessCodigo(Código):: double: Método encargado de calcular y devolver el fitness/aptitud del código.
  - obtenResultado(Código):: List<Integer>: Método responsable de obtener el resultado correspondiente al código recibido como parámetro comparado al código secreto (atributo de CalculoFitness).

1.1.13. Evolución

- Nombre de la classe:
  - Evolución
- Breve descripció de la classe:
  - Clase responsable de llevar a cabo la evolución de los códigos (usada en el entorno del algoritmo Genetic).

- Cardinalidad:
    - Uno por cada MáquinaGenetic.
- Descripción de los atributos:
    - numTorneos (Integer): Atributo que contiene el parámetro a tener en cuenta en el proceso de selección.
    - umbralRecombinación (double): Atributo correspondiente al umbral usado en el proceso de recombinación.
    - tasaMutación (double): Atributo correspondiente a la tasa de mutación usada en el proceso de mutación.
    - umbralPermutación (double): Atributo correspondiente al umbral usado en el proceso de permutación.
    - umbralInversión (double): Atributo correspondiente al umbral usado en el proceso de inversión.
- Descripción de las relaciones:
        - *– sin relaciones –*
- Descripción de los métodos:
    - evolucionaPoblacionCodigos(PoblaciónCódigos):: PoblaciónCodigos: Método encargado de causar la evolución de la población de códigos parámetro, siguiendo la ruta Selección, Recombinación, Mutación, Permutación y Inversión.
    - seleccionaCódigo(PoblaciónCódigos, int):: Código: Método encargado del proceso de selección de la población indicada como parámetro y con el parámetro del proceso numTorneos como segundo parámetro del método.
    - recombinaCódigos(Código,Código,double):: Código: Método encargado del proceso recombinación de los dos códigos parametrizados influenciada por el umbral de recombinación correspondiente al tercer parámetro.
    - mutaCódigo(Código, double):: Código: Método encargado del proceso de mutación del código parámetro influenciada por la tasa de mutación correspondiente al segundo parámetro.
    - permutaCódigo(Código, double):: Código: Método encargado del proceso de permutación del código parámetro influenciada por el umbral de permutación correspondiente al segundo parámetro.

- invierteCódigo(Código, double):: Código: Método encargado del proceso de inversión del código parámetro influenciada por el umbral de inversión correspondiente al segundo parámetro

1.1.14. ConfiguracionGenetic

- Nombre de la classe:
  - ConfiguracionGenetic
- Breve descripción de la clase:
  - Clase responsable de almacenar la configuración usada en el algoritmo Genetic (usada en el entorno del algoritmo Genetic).
- Cardinalidad:
  - Una única instància, clase estática.
- Descripción de los atributos:
  - númeroColores (Integer): Atributo correspondiente al número de colores diferentes de la configuración de la partida.
  - longitudCódigo (Integer): Atributo correspondiente a la longitud de los códigos de la partida.
  - maxGeneraciones (Integer): Atributo almacenador del número máximo de generaciones a generar en la evolución y/o llamados “maxSteps”.
  - númeroCódigosPoblación (Integer): Atributo que corresponde al número de códigos que deberán almacenar las poblaciones de códigos.
  - dificultad (Integer): Atributo que contiene la dificultad de la partida que influencia a las maquinas.
- Descripción de las relaciones:
  - – *sin relaciones* –
- Descripción de los métodos:
  - – *sólo contiene “getters” y “setters”* –

1.1.15. ConfiguracionPerfil

- Nombre de la classe:
  - ConfiguracionPerfil
- Breve descripció de la classe:
  - Clase responsable de almacenar la configuración usada en la clase Perfil.
- Cardinalidad:
  - Una para cada perfil
- Descripción de los atributos:
  - tema (Integer): Atributo correspondiente al tema seleccionado por el usuario.
- Descripción de las relaciones:
  - Relación de asociación con la clase “Perfil”: Permite que cada “Perfil” posea su configuración.

- Descripción de los métodos:
  – sólo contiene “getters” y “setters” –

1.1.16. Estadisticas

- Nombre de la classe:
  - Estadisticas
- Breve descripció de la classe:
  - Clase responsable de almacenar las estadísticas de las partidas jugadas por un usuario.
- Cardinalidad:
  - Una para cada perfil
- Descripción de los atributos:
  - partidasGanadas (Integer): Atributo correspondiente al número de partidasGanadas registradas por un usuario (Perfil).
  - partidasPerdidas (Integer): Atributo correspondiente al número de partidasPerdidas registradas por un usuario (Perfil).
  - mejorPuntuacion (double): Atributo que almacena la mayor puntuación conseguida en una partida por un jugador (Perfil).
  - idMejorPartida (Integer): Atributo que corresponde al id de la mejor partida jugada por un jugador (Perfil).
- Descripción de las relaciones:
  - Relación de asociación con la clase “Perfil”: Permite que cada “Perfil” posea sus Estadisticas.
- Descripción de los métodos:
  - anadirPartidaPerdida(): Método encargado de añadir 1 al contador de partidas perdidas por el “Perfil”
  - anadirPartidaGanada(): Método encargado de añadir 1 al contador de partidas ganadas por el “Perfil”
  - obtenPorcentajeVictoria(): Método encargado de devolver el porcentaje de victoria en base a las partidas ganadas y perdidas. La formula utilizada: (partidasGanadas / (partidasGanadas + PartidasPerdidas))*100
  - informaPuntuacion(double pPuntuacion, int IdPartida): Método encargado de comprobar si la puntuación recibida por parámetro supera la del objeto, en caso de que la supere actualiza la puntuación del objeto y se guarda el id de esa partida.

1.1.17. Perfil

- Nombre de la classe:
  - Perfil
- Breve descripció de la classe:

    - Otra de las clases “centrales” del diagrama. Se ocupa de guardar todo lo relacionado con el usuario.
- Cardinalidad:
    - n instancias
- Descripción de los atributos:
    - nombreUsuario (String): Atributo correspondiente al nombre de usuario. También es identificador de “Perfil”.
    - contrasena (String): Atributo correspondiente a la contraseña del usuario. Va asociada al nombre del usuario y se requiere de esta para iniciar sesión.
    - estadisticas (Estadisticas): Atributo almacenador de las estadisticas de juego de un usuario, es un puntero a esta.
    - partidas (ArrayList<Partida>): Atributo que corresponde a las partidas de un usuario, o sea, un vector de punteros a sus partidas.
    - fechaRegistro (LocalDate): Atributo que almacena la fecha de registro del usuario.
    - ultimaConexion (LocalDateTime): Atributo que almacena la fecha y hora de la ultima conexion del usuario
    - configPerfil (ConfiguracionPerfil): Atributo que almacena un puntero a la configuración de un usuario.
- Descripción de las relaciones:
    -
- Descripción de los métodos:
    - contrasenaCorrecta(String): Método que comprueba si la contraseña introducida como parámetro es la contraseña del usuario que intenta acceder.
    - obtenIdsPartidasSinAcabar(): Método que devuelve un Arraylist<Integer> con los ids de las partidas sin acabar del perfil
    - obtenIdsPartidasAcabadas(): Método que devuelve un Arraylist<Integer> con los ids de las partidas acabadas del perfil
    - obtenIdUltimaPartida(): Método que devuelve un integer que es el id de la última partida creada por el perfil
    - eliminarPartida(int idPartida): Método que elimina la partida del perfil con el id recibido por parámetro.
    - obtenPartida(int idPartida): Método que devuelve la instancia de Partida con el id recibido por parametro.

1.1.18. RankingGlobal

- Nombre de la classe:

    - RankingGlobal
- Breve descripció de la classe:
    - Clase responsable de almacenar el ranking global, es decir, las partidas con mayor puntuación entre todas.
- Cardinalidad:
    - Una única instància, clase estática.
- Descripción de los atributos:
    - ranking(RankingGlobal): Atributo que almacena el RankingGlobal, o sea, las mejores 10 partidas de todos los tiempos.
    - partidas (Partida[]): Atributo correspondiente a las partidas almacenadas en el ranking.
- Descripción de las relaciones:

  – *sin relaciones* –
- Descripción de los métodos:
    - anadirPartida(Partida): Método que añade una partida al RankingGlobal y actualiza el ranking de partidas.
    - reset(): Método que “resetea” el ranking de partidas.

1.1.19. RecordsMovimientos

- Nombre de la classe:
    - RecordsMovimientos
- Breve descripció de la classe:
    - Clase responsable de almacenar las partidas que se han acabado usando menos turnos.
- Cardinalidad:
    - Una única instància, clase estática.
- Descripción de los atributos:
    - recordsMovimients (RecordsMovimientos): Atributo que almacena el RecordsMovimientos, o sea, todas las configuraciones de partidas que se han solucionado con menos turnos.
    - records (Pair<Integer,String>[][]): Atributo que guarda una matriz que almacena en cada (fila,columna), correspondientes a longitudCodigo y numeroColores, un jugador y el número de turnos que ha necesitado para solucionar esa configuración.
- Descripción de las relaciones:

  – *sin relaciones* –
- Descripción de los métodos:
    - anadirPartida(String,int,int,int): Método que añade un usuario a RecordsMovimientos teniendo en cuenta la longitud del codigo

a romper, el numero de colores del juego y el numero de turnos que ha necesitado para ganar la partida.

1.2. Clases de presentación

1.2.1. ControladorPresentacio

1.2.2. ControladorVistaPantallaPrincipal

1.2.3. ControladorVistaSesion

1.2.4. BotonCorreccion

- Nombre de la classe:
  - BotonCorreccion
- Breve descripció de la classe:
  - Tipo de botón personalizado para que al ser pulsado cambie de estado, consta de tres estados:
    - Blanco: simboliza que no hay coincidencia
    - Gris claro: simboliza una coincidencia parcial
    - Negro: simboliza una coincidencia total
  - También consta de un atributo color, para facilitar la obtención del color y de dos atributos de coordenadas, para saber a qué posición pertenece ese botón.

1.2.5. BotonMuestra

- Nombre de la classe:
  - BotonMuestra
- Breve descripció de la classe:
  - Tipo de botón personalizado para que al ser pulsado envia a su padre su color para actualizar el color seleccionado.
  - También consta de un atributo color, para facilitar la obtención del color.

1.2.6. BotonPelota

- Nombre de la classe:
  - BotonPelota
- Breve descripció de la classe:
  - Tipo de botón personalizado para que al ser pulsado cambia de color al color seleccionado
  - También consta de dos atributos de coordenadas, para saber a qué posición pertenece ese botón.

1.2.7. CargarPartidaListener

- Nombre de la classe:
  - CargarPartidaListener
- Breve descripció de la classe:
  - Tipo de listener personalizado para que al pulsar un botón con este listener se cargue la partida a la que pertenece
  - Consta de un atributo idPartida para saber a qué partida pertenece y la vista de su padre para comunicarse con él.

1.2.8. EliminarPartidaListener

- Nombre de la classe:
  - EliminarPartidaListener
- Breve descripció de la classe:
  - Tipo de listener personalizado para que al pulsar un botón con este listener se elimine la partida a la que pertenece
  - Consta de un atributo idPartida para saber a qué partida pertenece y la vista de su padre para comunicarse con él.

1.2.9. Manual

- Nombre de la classe:
  - Manual
- Breve descripció de la classe:
  - Classe que al pasarle un número de página devuelve un string con el texto de esa página del Manual.

1.2.10. VistaConfiguracionPartida

- Nombre de la classe:
  - VistaConfiguracionPartida
- Breve descripció de la classe:
  - Vista que permite seleccionar la configuración deseada de la partida creada. Consta de distintos spinners:
    - Número de turnos
    - Tamaño de código
    - Número de colores
    - Nivel de ayuda
  - Cuando se trata de una partida de tipo codeMaker también muestra un desplegable para seleccionar la dificultado con las siguientes opciones:
    - Genetic - Facil
    - Genetic - Medio
    - Genetic - Dificil
    - Five Guess
  - También consta de un botón para volver atràs, dónde podràs seleccionar de nuevo el tipo de partida

1.2.11. VistaConfiguracionPerfil

- Nombre de la classe:
  - VistaConfiguracionPerfil
- Breve descripció de la classe:
  - Se trata de la vista la configuración del perfil, en esta se puede cambiar el tema utilizado. Se ve al hacer clic en “Configuracion” en VistaSesion. Contiene un título “CONFIGURACIÓN”, un tema, indicado por un label y un conjunto de botones que no tienen ninguna funcionalidad a parte de mostrar el conjunto de colores del tema seleccionado y 2 botones:
    - **Cambiar:** muestra la VistaTemas.
    - **Atras**: cierra la vista y vuelve a VistaSesion.

1.2.12. VistaEstadisticas

- Nombre de la classe:
  - VistaEstadisticas
- Breve descripció de la classe:
  - La vista estadísticas muestra las estadísticas del usuario, muestra:
    - Número de partidas jugadas
    - Número de partidas ganadas
    - Número de partidas perdidas

        - Porcentaje de victoria
    - También cuenta con un botón para salir.

1.2.13. VistaGuardarPartida

- Nombre de la classe:
    - VistaGuardarPartida
- Breve descripció de la classe:
    - Vista que muestra un mensaje de si quieres guardar la partida junto con dos botones:
        - Si: la partida se guarda
        - No: la partida es borrada del sistema

1.2.14. VistaIniciarSesion

- Nombre de la classe:
    - VistaIniciarSesion
- Breve descripció de la classe:
    - Se trata de la vista para iniciar sesión, la pantalla que se ve al hacer clic en “iniciar sesión” en VistaPantallaPrincipal. Contiene 2 campos para texto (text field), uno para el nombre de usuario y otro para la contraseña. Además contiene 2 botones:
        - **Enviar**: en caso de que los datos sean válido se inicia sesión y te lleva a VistaSesion, en caso contrario muestra un mensaje de error.
        - **Volver**: cierra la vista y te lleva a VistaPantallaPrincipal.

1.2.15. VistaIntroducirCodigo

- Nombre de la classe:
    - VistaIntroducirCodigo
- Breve descripció de la classe:
    - Se trata de la vista para introducir el código como CodeMaker, la pantalla se ve al hacer clic en “Empezar Partida” en VistaConfiguracionPartida jugando como CodeMaker. Contiene un título: “INTRODUCE EL CÓDIGO A ROMPER”, una cantidad variable de botones: el primer conjunto de botones es el código que proponemos, que tendrá un tamaño como lonigtudCodigo indicada en la vista anterior, además, el otro conjunto de botones, muestra los colores disponibles (dependiendo del tema seleccionado y el número de colores indicado en la vista anterior). Además hay otros 2 botones:
        - **Enviar**: envía el código a romper y empieza la partida jugando como CodeMaker.

- **Atras**: cierra la vista y te lleva a VistaSeleccionarTipoPartida.

1.2.16. VistaJugar

- Nombre de la classe:
  - VistaJugar
- Breve descripció de la classe:
  - Se trata de la vista para jugar. Te permite crear una partida nueva o cargar una existente. hay 3 botones:
    - **Crear Partida**: cierra la vista, y te lleva a VistaSeleccionarTipoPartida.
    - **Cargar Partida**: cierra la vista y te lleva a VistaPartidasGuardadas
    - **Atras**: cierra la vista y te lleva a VistaSesion.

1.2.17. VistaManual

- Nombre de la classe:
  - VistaManual
- Breve descripció de la classe:
  - Vista que muestra el manual, consta de un título (“MANUAL DE JUEGO”), una ventana de texto y tres botones
    - <- Botón para retroceder de página del manual
    - -> Botón para pasar de página del manual
    - Cerrar Manual: botón para cerrar al manual y volver a la vista anterior.

1.2.18. VistaPantallaPrincipal

- Nombre de la classe:
  - VistaPantallaPrincipal
- Breve descripció de la classe:
  Se trata de la vista inicial, la pantalla principal que se ve al iniciar el programa. Contiene un titulo “MASTERMIND” y 6 botones
  - **Registrarse**: te lleva a la vista de registro
  - **Iniciar Sesion**: te lleva a la vista para iniciar sesión
  - **Ver Records**: te lleva a la vista donde se muestran los records
  - **Ranking Global**: te lleva a la vista donde se muestra el ranking global
  - **Manual**: Te muestra el manual
  - **Salir**: cierra el programa

1.2.19. VistaPartidaCodeBreaker

- Nombre de la classe:
  - VistaPartidaCodeBreaker
- Breve descripció de la classe:
  - Se trata de la vista de la partida jugando como CodeBreaker, la pantalla que se ve al hacer clic en “Empezar Partida” en

VistaConfiguracionPartida o en el boton “Cargar” en VistaPartidasGuardadas. Contiene un panel de juego, con tantos botones como longitudCodigo*numeroTurnos para enviar una posible solucion cada turno. Contiene 3 botones a la izquierda y, tantos botones como longitudCodigo*numeroTurnos que hacen la corrección del código:

- **Enviar**: envía un código de colores y se ejecuta el siguiente turno en caso de que el código no sea el correcto.
- **PISTA**: muestra una ventana con un mensaje indicando una pista. En caso contrario, muestra una ventana diciendo que no quedan pistas.
- **SALIR**: abre una ventana diciendo si quieres guardar la partida. Cierra la vista y te lleva a la vista anterior (VistaPantallaPrincipal o VistaSesion).

1.2.20. VistaPartidaCodeMaker

- Nombre de la classe:
  - VistaCodeMaker
- Breve descripció de la classe:
  - Se trata de la vista de la partida jugando como CodeMaker, la pantalla que se ve al hacer clic en “Empezar Partida” en VistaConfiguracionPartida o en el botón “Cargar” enVistaPartidasGuardadas. Contiene un panel de juego, con tantos botones como longitudCodigo*numeroTurnos sin funcionalidad otra que mostrar los colores cada turno. Contiene 3 botones a la izquierda y, tantos botones como longitudCodigo*numeroTurnos para hacer la corrección del código:
    - **Enviar**: envía la corrección y en caso de que sea correcta, ejecuta el siguiente turno si quedan. En caso contrario, muestra una ventana diciendo que la corrección es incorrecta.
    - **AUTO CORREGIR**: hace una corrección automática y pasa al siguiente turno en caso de que te queden pistas. En caso contrario, muestra una ventana diciendo que no quedan pistas.
    - **Salir**: abre una ventana diciendo si quieres guardar la partida. Cierra la vista y te lleva a la vista anterior (VistaPantallaPrincipal o VistaSesion).

1.2.21. VistaPartidasGuardadas

- Nombre de la classe:

    - VistaPartidasGuardadas
- Breve descripció de la classe:
    - Vista que muestra las partidas guardadas del usuario, para cada partida muestra su ID, número de turnos, dificultad, nivel de ayuda, longitud de código, número de colores, tipo de partida y turno actual en el que se dejó. Adicionalmente cada partida cuenta con su propio botón de cargar y de eliminar la partida.
    - En caso de borrar todas las partidas te lleva a VistaSesion.

1.2.22. VistaRankingGlobal

- Nombre de la classe:
    - VistaRankingGlobal
- Breve descripció de la classe:
    - Se trata de la vista de los récords, la pantalla que se ve al hacer clic en “Ranking Global” en VistaPantallaPrincipal o en VistaSesion. Contiene un título “RECORDS”, una tabla donde se muestra la información del ranking global y 1 botón:
        - **Volver**: cierra la vista y te lleva a la vista desde donde se ha llamado (VistaPantallaPrincipal o VistaSesion).

1.2.23. VistaRecords

- Nombre de la classe:
    - VistaRecords
- Breve descripció de la classe:
    - Se trata de la vista de los récords, la pantalla que se ve al hacer clic en “Ver records” en VistaPantallaPrincipal o en VistaSesion. Contiene un título “RECORDS”, una tabla donde se muestra la información de los récords y 1 botón:
        - **Volver**: cierra la vista y te lleva a la vista desde donde se ha llamado (VistaPantallaPrincipal o VistaSesion).

1.2.24. VistaRegistrarse

- Nombre de la classe:
    - VistaRegistrarse
- Breve descripció de la classe:
    - Se trata de la vista para registrarse, la pantalla que se ve al hacer clic en “registrarse” en VistaPantallaPrincipal. Contiene 3 campos para texto (text field), de los cuales 2 son para la contraseña y su confirmación. Además contiene 2 botones:
        - **Enviar**: en caso de que los datos sean correctos se registra el usuario e inicia sesion, llevándote a

VistaSesión. En caso contrario muestra un mensaje de error.
- **Volver**: cierra la vista y te lleva a VistaPantallaPrincipal.

1.2.25. VistaSeleccionarTipoPartida

- Nombre de la classe:
  - VistaSeleccionarTipoPartida
- Breve descripció de la classe:
  - Se trata de la vista para seleccionar el tipo de partida, la pantalla que se ve al hacer clic en “crear partida” en VistaJugar. Contiene un título “SELECCIONA EL TIPO DE PARTIDA” y 3 botones:
    - **CODE-MAKER**: te lleva a VistaConfiguracionPartida.
    - **CODE-BREAKER**: te lleva a VistaConfiguracionPartida
    - **Volver Atras**: cierra la vista y te lleva a VistaJugar

1.2.26. VistaSesion

- Nombre de la classe:
  - VistaSesion
- Breve descripció de la classe:
  - Se trata de la vista de la sesión, la pantalla que se ve cuando se ha iniciado sesión. Contiene un título “MASTERMIND” y 7 botones:
    - **Jugar**: te lleva a VistaJugar.
    - **Configuracion**: te lleva a VistaConfiguracionPerfil
    - **Estadísticas**: te lleva a VistaEstadisticas
    - **Cerrar Sesion**: cierra sesion y te devuelve a VistaPantallaPrincipal
    - **Ranking Global**: te lleva a la vista donde se muestra el ranking global, en caso de estar vacío muestra un mensaje de error.
    - **Ver records**: te lleva a VistaRecords
    - **Manual**: Te muestra el manual

1.2.27. VistaTemas

- Nombre de la classe:
  - VistaTemas
- Breve descripció de la classe:
    - Se trata de la vista de los temas, la pantalla que se ve al hacer clic en “cambiar” en la vista ConfiguracionPerfil. Contiene un título “CONFIGURACIÓN”, 3 temas, cada uno indicado por un label y un conjunto de botones que

no tienen ninguna funcionalidad a parte de mostrar el conjunto de colores del tema. Hay 4 botones:

- **Seleccionar tema (x3):** selecciona el tema indicado, cierra la vista y vuelve a VistaConfiguracionPerfil.
- **Atras**: cierra la vista y vuelve a VistaConfiguracionPerfil.

1.3. Clases de persistencia

1.3.1. ControladorPersistencia

- Nombre de la classe:
  - ControladorPersistencia
- Breve descripció de la classe:
  - Clase controladora de la capa de persistencia encargada de la creación, guardado y borrado de datos mediante serialización para permitir la existencia de persistencia en el sistema.
- Cardinalidad:
  - Singleton, uno único.
- Descripción de los atributos:
  - ctrl (ControladorPersistencia): Instancia singleton.
  - fileInputStream (FileInputStream): Variable contenedora del contenido de datos persistentes crudos.
  - fileOutputStream (FileOutputStream): Variable contenedora del contenido de datos persistentes crudos a escribir.
  - objectInputStream (ObjectInputStream): Variable contenedora del contenido de datos persistentes en forma de objeto leídos.
  - objectOutputStream (ObjectOutputStream): Variable contenedora del contenido de datos persistentes en forma de objeto a escribir.
- Descripción de las relaciones:
  - Relación de asociación con “RankingGlobal”: Indica el archivo contenedor de la información del RankingGlobal persistente.
  - Relación de asociación con “Perfiles”: Indica el archivo contenedor de la información de los perfiles del sistema persistentes.
  - Relación de asociación con “RecordsMovimientos”: Indica el archivo contenedor de la información de los RecordsMovimientos del sistema persistentes.
- Descripción de los métodos:
  - El controlador de persistencia contiene una función de lectura y escritura para cada relación descrita, encargada de leer datos transformándolos a tipo Objeto genérico y de escribir los objetos genéricos recibidos como parámetro del archivo en disco en cuestión mediante serialización.