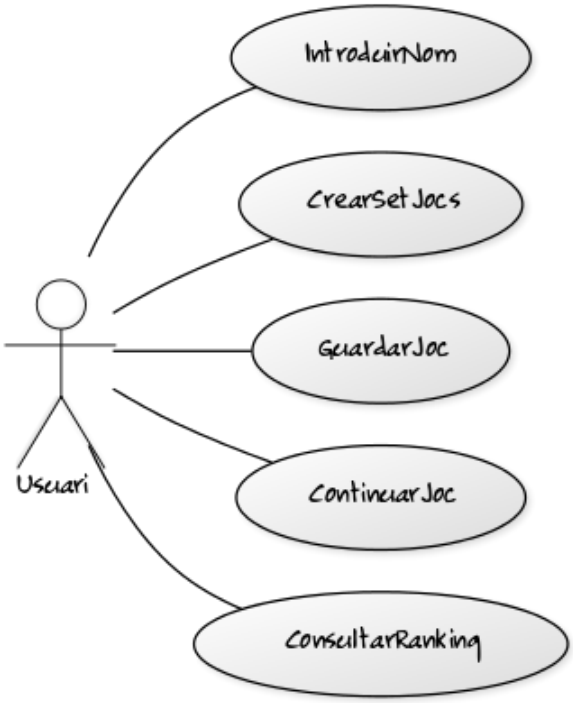
Documentació

Diagrama de casos d'ús



Cas d'ús: AltaNom

Actor: Usuari

Comanda al main: "INTRODUEIXNOM"

Descripció:

El sistema demana el nom a l'usuari: "introdueix nom". L'usuari l'introdueix i el llegeix un objecte de la classe *Scanner*, que l'assigna a l'atribut *name*.

Cas d'ús: JugarSet

Actor: Usuari

Comanda al main: "CREARSETJOCS"

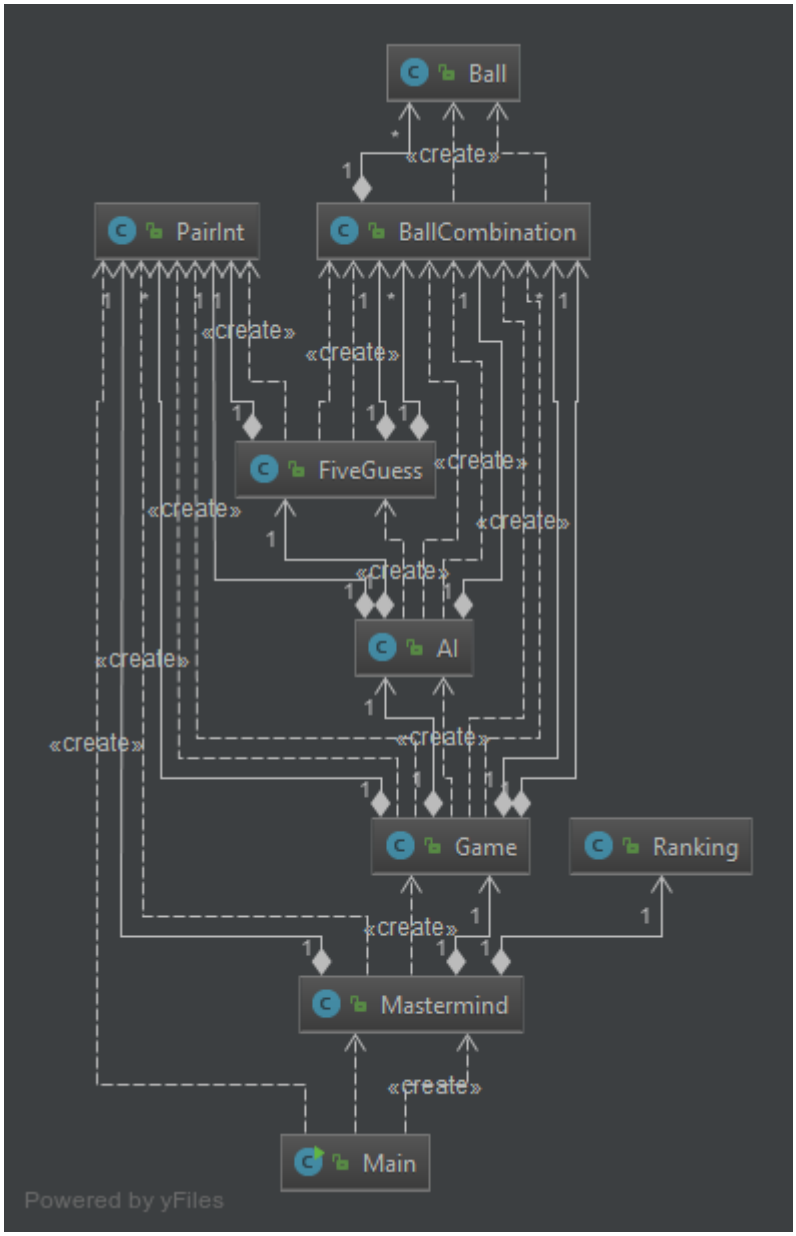
Descripció:

El sistema demana el nombre de partides i dificultat a l'usuari: "Introdueix nombre de partides i dificultat". L'usuari els introdueix i els llegeixen dos objectes *Scanner.* Assignem el nombre de partides dividit entre 2 a l'atribut gamesNeeded, i assignem dificultat a l'atribut difficulty. Aquest atribut el passem a l'objecte m de la classe Mastermind per fixar el mode de dificultat del set de partides (atributs n, f i nColors de Mastermind). Inicialitzem més atributs de m (nGuesses, score, minGuesses.a=100,

minGuesses.b=100). Entrem en un loop al qual serem dins sempre que ni score.a ni score.b siguin diferents de gamesNeeded. Neguem humanPlaying, i si és fals (si toca que jugui la IA) el sistema demana la solució del següent joc: “Escriu la solució del joc”. Aquest codi solució es passa a m amb la funció setSolution, que n’actualitza l’atribut. Guardem les estadístiques del joc a m i s’imprimeixen per pantalla. Finalment, un cop finalitzat el loop, li passem el nom del set a m i demanem que guardi l’estat cridant save_params_to_txt(name). Finalment el sistema imprimeix: “Set jugat”.

Els casos d’ús restants fan referència a la capa de dades i no els tenim encara implementats.

Diagrama UML



Descripció de les classes del joc:

Mastermind: Fàbrica de partides de mastermind que també actua com a controlador de la capa de domini, carregant els paràmetres de configuració i actualitzant els valors que més tard aniran a la capa de dades. El control de partides el fa mitjançant tirades individuals que assigna al usuari o a l'actor IA.

Game: Classe que conté el taulell de la partida, on es guarden totes les tirades d'aquesta amb les respostes en boles blanques i negres que han generat respecte la solució.

AI: Es la encarregada de fer els càlculs per enviar una tirada nova a Game mitjançant la classe FiveGuess i d'establir la solució de la partida quan AI fa de codemaker.

BallCombination: Struct que conté el nombre de boles que tindrà la combinació, l'offset de colors que es podran representar i una array amb les boles que componen la combinació.

Ball: Classe que conté un únic atribut corresponent al color d'una bola.

FiveGuess: Conté totes les operacions necessàries per a calcular una nova tirada de la partida mitjançant l'algorisme fiveguess, com les funcions de filtre i minmax.

PairInt: Struct que conté dos valors enters. S'usa per a càlculs de resultats de combinacions i puntuacions de jugadors.

Ranking: S'ocupa de interactuar amb la part de la capa de dades corresponent a la informació del rànking(crear dades i modificar-les).

Relació de les classes implementades

Totes les classes han sigut implementades pels tres membres del grup ja que es van deixar quasi fetes durant les reunions per a decidir l'estructura de les classes del joc. Després els tres membres hem anat modificant totes les classes segons era oportú. Hi ha parts que han sigut treballades amb més èmfasi per alguns dels membres, les quals, son les següents:

Mario: Algoritme, Ranking, debugging

Jordi: Disseny de les classes

Raul: Jocs de proves, drivers, debugging