MEMORIA

Apartado 5

Cuestiones

1) Imagina que te piden realizar otro juego de tablero (por ejemplo, ajedrez o damas). ¿Qué clases o interfaces podrías reutilizar? ¿Habría que añadir algún método adicional?

   Como interfaces podríamos perfectamente añadir las 3 interfaces. Ya que ambos juegos utilizan los mismos elementos, tablero, fichas y celdas.

   Simplemente habría que moldear el tamaño del tablero con una clase que implementase el tablero sobre el que se quiere jugar el juego.

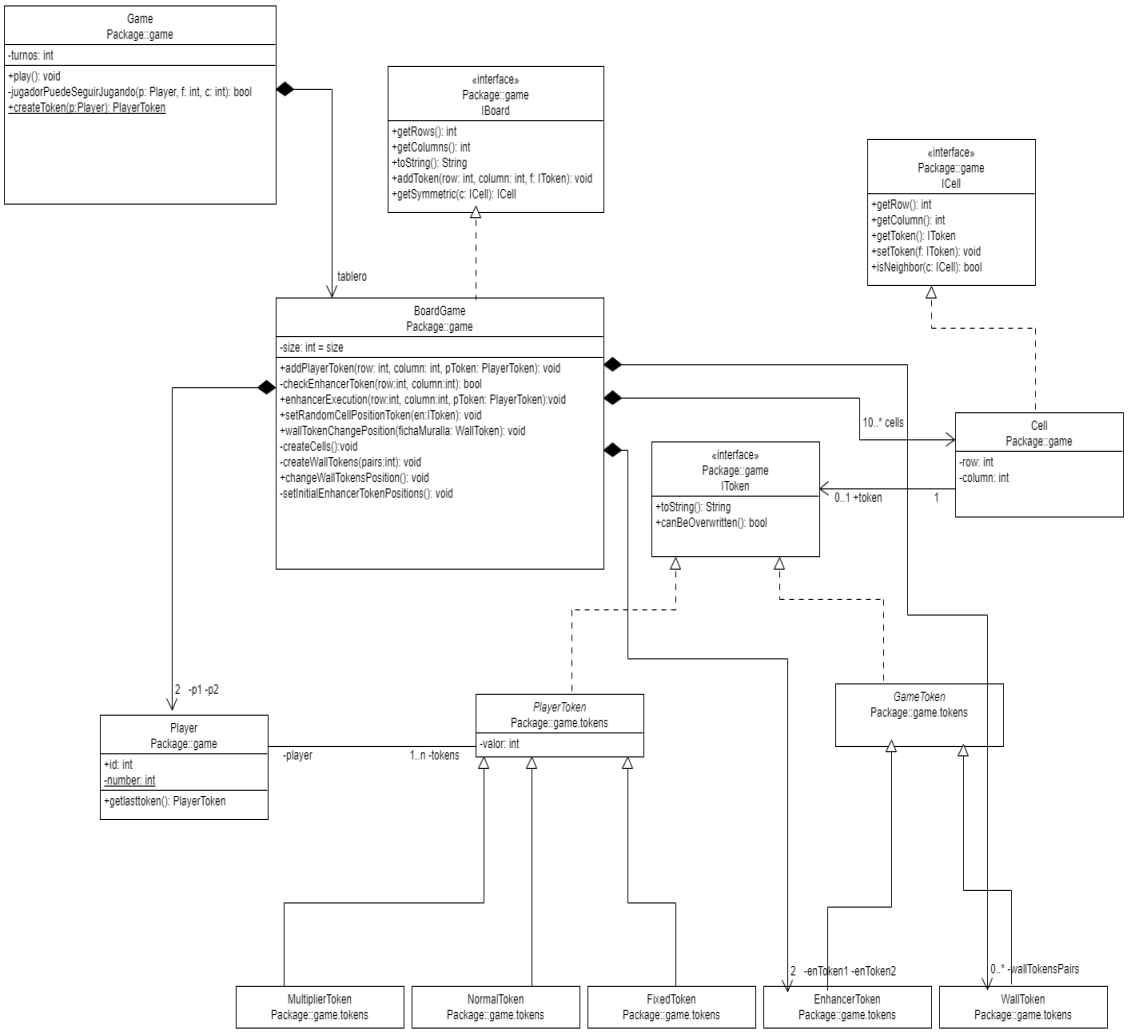
   Respecto a las fichas se crearía 1clase abstracta que implementase la interfaz fichas, y todas la categorías distintas de fichas heredarían de la principal con una funcionalidad principal y con métodos de clase que restrinja sus movimientos. con las características correspondientes y todas podrían ser sobrescritas (se pueden comer entre ellas). Y añadiría métodos específicos con los movimientos que pueden realizar cada tipo de ficho.

   Respecto a las celdas, habría celdas blancas y negras pero no son distinguibles entre ellas a nivel de funcionalidad en principio. Por tanto, lo dejaría como una clase.

2) Supón que creamos una ficha adicional cuyo valor es temporal. Es decir, cuando se coloca por primera vez tiene un valor de 3 puntos pero después de 3 turnos, su valor vuelve a ser 1. ¿Cómo incluirías esta nueva funcionalidad? No hace falta programar nada, solo indicar cómo hacerlo.

   Como método de clase dentro de la clase de esa nueva ficha, además de asignarle un atributo de tipo entero turnos que la clase Game actualizase su contador de turnos después de cada turno, llamando al método de clase de esa nueva ficha.

Diagrama UML



Planteamiento

Como podemos observar, tenemos una clase general que guarda una clase Tablero como atributo. Esa clase, además va a controlar los turnos del juego y va a controlar que se cumplen los requisitos generales para seguir la partida con el método play().

Por otra parte, el tablero va a tener como atributos, los dos jugadores que pueden jugar sobre el tablero, que jugador se trata de otra clase aparte. También va a tener como atributo un ArrayList de objetos de clase celda. Las cuales implementan la interfaz ICell que nos venía dada.

Por otra parte, como el tablero en este caso, solo puede tener 2 fichas potenciadoras, hemos guardado como atributo dos fichas potenciadoras, en vez de fichas de juego, para poder controlarlo mejor. Y respecto a las fichas muralla, como solo puede tener pares de fichas muralla, lo que hemos hecho es que tablero tenga un ArrayList de arrays de tamaño 2 (pares) de Celdas que contengan los pares en posiciones simétricas de esas fichas murallas, así nos es más fácil poder controlarlo. Ambas fichas heredan de la clase abstracta fichas de juego (que no pertenecen a ningún jugador), que a su vez esta implementa la interfaz IToken.

Por otro lado estarían las fichas de jugador, que son distintas porque tienen como atributo un jugador y tienen un valor asociado (ese valor ya depende de que tipo de ficha se tratan), cada tipo de ficha es una clase que hereda de la abstracta de la clase FichaJugador , que a su vez esta implementa también la interfaz IToken.

También debemos mencionar que el tablero implementa la interfaz IBoard, además de todos los métodos que hemos añadido para facilitar el control de la partida, tanto privados como públicos según era necesario o no.