Ajedrez

El objetivo del trabajo práctico fue desarrollar un juego de ajedrez que se pudiera jugar de a uno o dos jugadores con la posibilidad de revertir jugadas. El programa realizado intenta aplicar el patrón MVC (Model-View-Controller) para implementar un juego de ajedrez con una interfaz gráfica amigable al usuario separada estructuralmente de la lógica interna del juego.

Controlador:

Dentro del paquete *logica*, hay una clase llamada Juego. Ésta clase funciona como la intermediaria entre la interfaz gráfica (View) y la lógica interna (Model). Como tal, provee una gama amplia de funcionalidades para cualquier vista que quiera ser implementada sobre ésta. Es la responsable de mantener el estado del juego, como de quién es el turno, si hubo jaque mate, qué jugadas se hicieron, etc. Así, todos los pedidos que se realicen desde la interfaz son analizados por el controlador y potencialmente enviados a la lógica interna para ser procesados.

Modelo:

A diferencia del controlador, el modelo está compuesto por varias clases, todas contenidas en el paquete *logica*. La clase principal es *Tablero*. Como su nombre lo indica, contiene la representación física del tablero. Las piezas del juego vienen representadas en una jerarquía: cada tipo de pieza tiene su propia clase (*Peon, Caballo, Rey, Dama, Alfil* y *Torre)*, pero todas ellas son subclases de la clase abstracta *Pieza*. La clase *Tablero* representa al tablero físico con una matriz de 8x8 casilleros (instancias de la clase *Casillero*). Cada uno de estos contiene una referencia a la pieza que lo ocupa (o a *null* si está vacío). Esto destaca un rasgo importante de la lógica que define varios aspectos importantes: **el tablero es el único que sabe dónde están las piezas**. Las piezas per se no conocen su ubicación en el tablero (de hecho, ni siquiera saber si están o no en el tablero). Esto le da una importante responsabilidad al tablero. Sin embargo, el comportamiento de las piezas no deja de estar (casi) en su totalidad encapsulado en las propias piezas. Pero en las cuestiones en las que intervienen dos o más piezas (jaque o enroque), parte de la responsabilidad quedó puesta en el tablero. Así, por ejemplo, las piezas son las responsables de expresar sus movimientos (a través de listas de objetos de tipo *Movimiento*), pero el rey no es capaz de decidir por sí solo si está o no en jaque.

Existen en el paquete *logica* algunas clases que son compartidas por todos los módulos, ya que son universales para cualquier parte del programa y son una vía de codificación de la información para viajar del modelo hacia la vista. Éstas son *ColorPieza*, *NombrePieza*, *NombreColorPieza* y *PosicionAjedrez*. Otras clases son exclusivas de la lógica, como *PosicionTablero*, que responde a la representación interna del tablero, a diferencia de *PosicionAjedrez*, que responde a la representación convencional de posiciones en el ajedrez (ej: “e3”).

Existe otra clase llama *Jugada*. Cada objeto de esta clase es la representación de una jugada. Sirve entre otras cosas para revertir las jugadas que se van haciendo, al ir el controlador guardándolas en un Stack.

Vista:

/\* JavaFX blah blah blah… \*/

Inteligencia:

Además de toda la representación del juego, existe una inteligencia artificial que permite al usuario jugar “contra la máquina”. Ésta inteligencia tiene dos modos. Inicialmente, consta de un árbol de jugadas cargado desde un archivo XML mediante el farmework *Jdom*. Este árbol define series de jugadas que la inteligencia puede o no ir haciendo en base a la jugada del adversario. Una vez agotadas las posibilidades del árbol, la inteligencia entra en modo “pensar”, donde debe elegir por sí sola la jugada a realizar. Para ello, aplica un algoritmo con regla de decisión *minimax* y una heurística basada en los puntajes clásicos de las piezas.