**User Manual**

**Ryan Tibbetts**

**Introduction**

The following is a step by step instructional on how to use **ChessEnginePrototype**.

**PreRequisites**

Make sure the program contains all necessary files. I shall list these files in a non-specific order:

* ABChess.java
* Moves.java
* Rating.java
* Userinterface.java
* ChessPieces.png

ChessPieces.png should be the in the same folder as src, not in src.

The Main is in ABChess.java

**Step 1, Run Program**

A chess interface should pop up along with a window asking you to pick your side. If you exit this window without picking you will automatically be set as white. If you chose black your king will be left side although your color will not change. Color is merely aesthetic and this does not affect the performance of the code, get over it.

**Step 2, Refresh Interface**

Upon selecting or not selecting a side you will need to single click the chessboard interface, this will set your game up proper assuming your board was jumbled or incorrect, if not then neglect this step.

**Step 3, Moving Pieces**

To move a piece, simply click and drag. Your piece will not make moves it is incapable of performing. These rules are based on the rules of chess. Note that castling and en passant are not available in this version of the application.

**Step 4, Winning and Losing**

**Win**

Upon winning the game, the computer will return a random board on your interface. Do not be afraid this is merely the programs way of accepting defeat as it struggles to comprehend its inability to produce a move. This was never fixed merely for my own personal amusement of the programs panic. There is no reset button and you must rerun the program.

**Lose**

Often times you will not actually lose, but be put in a position of back and forth stalemate. Stalemate will not initiate and you must use your own power of intuition and admit defeat. If you have determined that you have lost, simply exit the program. To play again you must run the program again.

**Step 5 Change Difficulties**

If you desire to play a harder level you may go into the code to change the algorithm’s depth search. First go into “ABChess.java”; you will find a static integer called “globalDepth,” it will be set to 4. The higher the set depth is, the harder the game will be. Any depth past 4 will likely run slower. It is ill advised to set the depth past 6, however the engine will calculate as many depths as you desire… in theory… this is not actually tested since I don’t have a computer fast enough to run some of the deeper depths.

See next page for **ChessEngine** package informal running operations.

**Informal Explanation**

This is an informal explanation for how to run the debug on ChessEngine:

* You will find a command in UserInterface.java under the main called BoardGenerdation.importFen(“rnbqkbr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq – 0 1”).
* Lowercase values are black, uppercase are white.
* Each “/” is a rank on the board.
* To make a blank space, you can add a number 1-8 to represent the amount of spaces.
* Example: pp4pp is 2 pawns 4 spaces and 2 pawns.
* R = rook, N = knight, B = bishop, K = king, Q = queen, P = pawn
* w = whites turn, b in that same position = blacks turn
* KQkq represents if castling is available either kingside or queenside for their respective color. To remove it’s availability replace the letter with a dash. Do not use multiple dashes unless both queensides are unavailable or both kingsides are unavalible.
* The dash represents En Passant possible as represented by their respective long file between 1-8 (see Moves.java to see these files)
* I never actually determined what the 0 1 was. It was inconsequential. But can be replaced with a – not for each, but for both together.
* If the parameters I have given you are not met, perft will not perfom correctly.
* Upon running the program a fixed GUI will pop up and the command line will produce a text board set with importFEN()’s interface.
* The program will then play each move to get to where importFEN() is set too at a depth 6 search. If you do not alter the importFEN() interface then the program will search all nodes for the given depth and return a total node value. This is used for debugging.
* Depth can be changed in “Perft.Java” by changing the number of static integer “perftMaxDepth”
* All accountable nodes for depth six and lower are listed in “Perft Data.txt”
* There are several preset importFEN()’s in “Perft Data.txt” and you are welcome to try these.

The ChessEngine package does nothing more at this moment in time, but my research for it is in the paper.

**Enjoy playing chess!**