ChessEngine.java

The main class for checking chess moves and interacting with the Stockfish chess engine

There are 8 public methods (as of 5/4/15 18:00), but only 3 really significant.

To use the engine, the sequence is,
First instantiate a chess engine

```
ChessEngine myEngine = new ChessEngine(PlayColour.BLACK) (or White)
```

Then repeat the sequence:

1. When the user selects a piece, call `getLegalMoves(Square)` to find where they can move the piece TO
2. When the user selects a (legal) destination, call `makeAMove (Square, Square)`
3. Then call `Move move = engineMove()`. This returns the move made, so the board can be updated.

Thats it! There are other methods to help, like `getGameFen()` returns a FEN game description (look it up in Wikipedia).
`whoseMove()` removes any doubt.
`getPieceAt(Square square)` returns the piece at the square, or null. The class also keeps move history, new methods will be needed to return this history for display.

The Doxygen docs are fully complete (I believe)

ChessEngine uses three libraries and an executable image:

- ICTK which provides many chess classes and methods, including ChessBoard, ChessPiece, Piece, Square etc. This could fill out most of our Chess game if we chose - anyway I use it and some of the methods return objects from this library.
- stockfish.java is a small library class designed to manage the interface to the StockFish chess engine, it manages the standard-Input/Output streams. It is simply included in the src directory, and in fact has been edited to be part of the swantech package.

- JUnit, the unit testing framework. Directory Tests/swantech contain test classes. ChessTestSuite simply runs each test class (and new tests should be added to this).
- In addition, engine/stockfish is an executable StockFish engine, that communicates via standard-IN and standard-OUT. This is an executable image, they are available for OS X , Linux and Windows (but not IOS I don't think, that will cause problems).

Acknowledgements

All the libraries, source code, utilities and programs used have been released under open-source licenses that permit us to use, modify and extend as we wish. Thanks are due to:

- Jason Varsoke for ICTK,
  https://github.com/jvarsoke/ictk/tree/master/src/samples#simplepgndemo
  http://ictk.sourceforge.net/docs/current/
  http://ictk.sourceforge.net/#downloads

- Stockfish thanks to Tord Romstad, Marco Costalba, and Joona Kiiski.
  https://stockfishchess.org/

- Rahul for the Stockfish Java interface, at
  http://rahular.com/stockfish-port-for-java/

- the Junit community at
  http://junit.org/

Still to be done:

1. History. The class keep history, a simple method is needed to return the history as a string.
2. User profile? Probably no need. The Chess Engine users are set up as *Boris Spasky* and *Bobby Fischer*, but this is never seen.
3. Castling, en-passant takes, promotion. The ICTK library and Stockfish supports all of these but I have provided no interface yet.
4. Not sure if the current methods provide sufficient coverage for check and check mate. The Move object returned by the two move methods itself has methods such as `isCheck()` , `isCheckMate()` etc, I think that will be enough. eg:

```
Move m = engineMove();
if (m.isCheckMate()) {
    // game over
    } else {
    // move piece to new square:
    Square s = m.getPiece().getSquare();
```