

ToldiChess

Generated by Doxygen 1.9.1

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 Class Documentation	5
3.1 pieces.Bishop Class Reference	5
3.1.1 Detailed Description	5
3.1.2 Member Function Documentation	5
3.1.2.1 validMove()	5
3.2 game.Board Class Reference	6
3.2.1 Detailed Description	6
3.2.2 Constructor & Destructor Documentation	6
3.2.2.1 Board() [1/3]	6
3.2.2.2 Board() [2/3]	6
3.2.2.3 Board() [3/3]	7
3.2.3 Member Function Documentation	7
3.2.3.1 castling()	7
3.2.3.2 executeMove()	7
3.2.3.3 getFen()	8
3.2.3.4 inCheck()	8
3.2.3.5 isMate()	8
3.2.3.6 isStalemate()	9
3.2.3.7 validMove() [1/2]	9
3.2.3.8 validMove() [2/2]	9
3.3 game.Fen Class Reference	10
3.3.1 Detailed Description	10
3.3.2 Constructor & Destructor Documentation	10
3.3.2.1 Fen()	10
3.3.3 Member Function Documentation	11
3.3.3.1 addTurn()	11
3.3.3.2 getCastling()	11
3.3.3.3 getCastlingAt()	11
3.3.3.4 getEnpassan()	11
3.3.3.5 getPiece()	12
3.3.3.6 getPlayer()	12
3.3.3.7 getTile()	12
3.3.3.8 getTiles()	13
3.3.3.9 setCastling()	13
3.3.3.10 setEnpassan()	13
3.3.3.11 setPlayer()	14
3.3.3.12 setTile()	14

3.3.3.13 toString()	14
3.4 swing.GameGUI Class Reference	14
3.4.1 Detailed Description	15
3.4.2 Constructor & Destructor Documentation	15
3.4.2.1 GameGUI()	15
3.4.3 Member Function Documentation	15
3.4.3.1 getState()	15
3.4.3.2 setState()	15
3.5 pieces.King Class Reference	16
3.5.1 Detailed Description	16
3.5.2 Member Function Documentation	16
3.5.2.1 validMove()	16
3.6 pieces.Knight Class Reference	17
3.6.1 Detailed Description	17
3.6.2 Member Function Documentation	17
3.6.2.1 validMove()	17
3.7 util.LogComparator Class Reference	18
3.7.1 Detailed Description	18
3.8 util.Logger Class Reference	18
3.8.1 Detailed Description	19
3.8.2 Constructor & Destructor Documentation	19
3.8.2.1 Logger() [1/2]	19
3.8.2.2 Logger() [2/2]	19
3.8.3 Member Function Documentation	19
3.8.3.1 configEnd()	19
3.8.3.2 drop()	20
3.8.3.3 get()	20
3.8.3.4 getLast()	20
3.8.3.5 getMoves()	20
3.8.3.6 getName()	21
3.8.3.7 put()	21
3.8.3.8 save()	21
3.9 swing.Logs Class Reference	21
3.9.1 Detailed Description	22
3.9.2 Constructor & Destructor Documentation	22
3.9.2.1 Logs()	22
3.9.3 Member Function Documentation	22
3.9.3.1 getState()	22
3.9.3.2 initNames()	22
3.9.3.3 setState()	22
3.10 swing.MainMenu Class Reference	23
3.10.1 Detailed Description	23

3.10.2 Constructor & Destructor Documentation	23
3.10.2.1 MainMenu()	23
3.10.3 Member Function Documentation	23
3.10.3.1 run()	24
3.11 pieces.Pawn Class Reference	24
3.11.1 Detailed Description	24
3.11.2 Member Function Documentation	24
3.11.2.1 validMove()	24
3.12 pieces.Piece Class Reference	25
3.12.1 Detailed Description	25
3.12.2 Member Function Documentation	25
3.12.2.1 equals()	25
3.12.2.2 getColor()	26
3.12.2.3 getType()	26
3.12.2.4 validMove()	26
3.13 util.PieceKey Class Reference	27
3.13.1 Detailed Description	27
3.14 util.PieceType Enum Reference	27
3.14.1 Detailed Description	27
3.15 util.Pos Class Reference	28
3.15.1 Detailed Description	28
3.16 pieces.Queen Class Reference	28
3.16.1 Detailed Description	29
3.16.2 Member Function Documentation	29
3.16.2.1 validMove()	29
3.17 pieces.Rook Class Reference	29
3.17.1 Detailed Description	30
3.17.2 Member Function Documentation	30
3.17.2.1 validMove()	30
3.18 util.State Enum Reference	30
3.18.1 Detailed Description	31
3.19 swing.SwTile Class Reference	31
3.19.1 Detailed Description	31
3.20 util.Team Enum Reference	31
3.20.1 Detailed Description	32
3.21 game.Tile Class Reference	32
3.21.1 Detailed Description	32
3.21.2 Constructor & Destructor Documentation	32
3.21.2.1 Tile() [1/3]	32
3.21.2.2 Tile() [2/3]	33
3.21.2.3 Tile() [3/3]	33
3.21.3 Member Function Documentation	33

3.21.3.1 getPiece()	33
3.21.3.2 getPos()	34
3.21.3.3 setPos()	34

Index	35
------------------------------	-----------

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

game.Board	6
JButton	
swing.SwTile	31
JFrame	
swing.MainMenu	23
JPanel	
swing.GameGUI	14
swing.Logs	21
util.PieceKey	27
util.PieceType	27
Runnable	
swing.MainMenu	23
util.State	30
Comparator	
util.LogComparator	18
Serializable	
game.Fen	10
game.Tile	32
pieces.Piece	25
pieces.Bishop	5
pieces.King	16
pieces.Knight	17
pieces.Pawn	24
pieces.Queen	28
pieces.Rook	29
util.Logger	18
util.Pos	28
util.Team	31

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

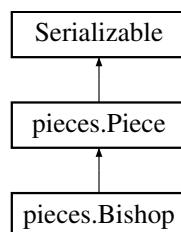
pieces.Bishop	5
game.Board	6
game.Fen	10
swing.GameGUI	14
pieces.King	16
pieces.Knight	17
util.LogComparator	18
util.Logger	18
swing.Logs	21
swing.MainMenu	23
pieces.Pawn	24
pieces.Piece	25
util.PieceKey	27
util.PieceType	27
util.Pos	28
pieces.Queen	28
pieces.Rook	29
util.State	30
swing.SwTile	31
util.Team	31
game.Tile	32

Chapter 3

Class Documentation

3.1 pieces.Bishop Class Reference

Inheritance diagram for pieces.Bishop:



Public Member Functions

- **Bishop** ([Team](#) color)
- boolean **validMove** ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Additional Inherited Members

3.1.1 Detailed Description

Classic chess [Bishop](#).

3.1.2 Member Function Documentation

3.1.2.1 validMove()

```
boolean pieces.Bishop.validMove (
    Pos from,
    Pos to,
    Fen fen )
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To positon.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

Reimplemented from [pieces.Piece](#).

The documentation for this class was generated from the following file:

- `src/pieces/Bishop.java`

3.2 game.Board Class Reference

Public Member Functions

- [Board](#) ()
- [Board](#) (String fen)
- [Board](#) ([Board](#) b)
- boolean [validMove](#) ([Pos](#) from, [Pos](#) to)
- boolean [validMove](#) ([Pos](#) from, [Pos](#) to, boolean checkcheck, [Team](#) asTeam)
- int [castling](#) ([Pos](#) from, [Pos](#) to)
- void [executeMove](#) ([Pos](#) from, [Pos](#) to)
- boolean [inCheck](#) ([Team](#) color)
- boolean [isStalemate](#) ()
- boolean [isMate](#) ()
- [Fen](#) [getFen](#) ()

3.2.1 Detailed Description

[Board](#) class, used in executing and checking of the moves. Also checks the Mate and Stalemate states.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 [Board](#)() [1/3]

```
game.Board.Board ( )
```

Default [Board](#) Constructor. Creates a [Board](#) with the standard chess setup.

3.2.2.2 [Board](#)() [2/3]

```
game.Board.Board (
    String fen )
```

Custom String [Board](#) Constructor. Creates a [Board](#) with the given FEN

Parameters

<i>fen</i>	The fen which will be the state of the board, in String format.
------------	---

3.2.2.3 Board() [3/3]

```
game.Board.Board (
    Board b )
```

[Board](#) copy constructor.

Parameters

<i>b</i>	The copyable board.
----------	---------------------

3.2.3 Member Function Documentation**3.2.3.1 castling()**

```
int game.Board.castling (
    Pos from,
    Pos to )
```

Returns which castling move a player is performing, and it's validity.

Parameters

<i>from</i>	From position.
<i>to</i>	To Position.

Returns

0 - White Kingside, 1 - White Queenside, 2 - Black Kingside, 3 - Black Queenside, -1 - None.

3.2.3.2 executeMove()

```
void game.Board.executeMove (
    Pos from,
    Pos to )
```

Executes a move, updating the fen in the process.

Parameters

<i>from</i>	From position.
<i>to</i>	To position.

3.2.3.3 getFen()

```
Fen game.Board.getFen ( )
```

Returns the board's fen.

Returns

The fen of the [Board](#).

3.2.3.4 inCheck()

```
boolean game.Board.inCheck (
    Team color )
```

Checks if there is a Check on the board for a given player.

Parameters

<i>color</i>	The players color which will be checked.
--------------	--

Returns

True if in check.

3.2.3.5 isMate()

```
boolean game.Board.isMate ( )
```

Checks if a mate has occurred for the current player.

Returns

True if it has.

3.2.3.6 isStalemate()

```
boolean game.Board.isStalemate ( )
```

Checks if a stalemate has occurred.

Returns

True if it has.

3.2.3.7 validMove() [1/2]

```
boolean game.Board.validMove (
    Pos from,
    Pos to )
```

Move validator with the default option of checking for castling and the moving player being the one currently moving.

Parameters

<i>from</i>	From position.
<i>to</i>	To position.

Returns

True if the move is valid.

3.2.3.8 validMove() [2/2]

```
boolean game.Board.validMove (
    Pos from,
    Pos to,
    boolean checkcheck,
    Team asTeam )
```

Customisable move validator.

Parameters

<i>from</i>	From position.
<i>to</i>	To position.
<i>checkcheck</i>	If true, will check not to move into checks.
<i>asTeam</i>	Color of the moving pieces.

Returns

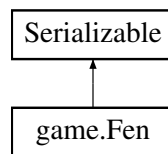
True if move is valid.

The documentation for this class was generated from the following file:

- `src/game/Board.java`

3.3 game.Fen Class Reference

Inheritance diagram for game.Fen:



Public Member Functions

- [Fen](#) (String fen)
- String [toString](#) ()
- [Piece](#) [getPiece](#) ([Pos](#) pos)
- [Tile](#)[][] [getTiles](#) ()
- [Tile](#) [getTile](#) ([Pos](#) pos)
- void [setTile](#) ([Tile](#) tile)
- [Team](#) [getPlayer](#) ()
- void [addTurn](#) ()
- void [setPlayer](#) ([Team](#) player)
- [Pos](#) [getEnpassan](#) ()
- void [setEnpassan](#) ([Pos](#) enpassan)
- void [setCastling](#) (int inx, boolean value)
- boolean [getCastlingAt](#) (int inx)
- boolean[] [getCastling](#) ()

3.3.1 Detailed Description

[Fen](#) class used to store the state of the board at any given moment. Serializable because of io storage.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Fen()

```
game.Fen.Fen (  
    String fen )
```

Creates a custom fen from a string.

Parameters

<i>fen</i>	The string of the fen.
------------	------------------------

3.3.3 Member Function Documentation

3.3.3.1 addTurn()

```
void game.Fen.addTurn ( )
```

Increments the turns by 1.

3.3.3.2 getCastling()

```
boolean [ ] game.Fen.getCastling ( )
```

Returns the array of castling booleans.

Returns

Castling booleans.

3.3.3.3 getCastlingAt()

```
boolean game.Fen.getCastlingAt (
    int inx )
```

Returns the castling availability at the current index.

Parameters

<i>inx</i>	0-WKingside, 1-WQueenside, 2-BKingside, 3-BQueenside.
------------	---

Returns

True if its available.

3.3.3.4 getEnpassan()

```
Pos game.Fen.getEnpassan ( )
```

Return the EnPassan position on the board.

Returns

EnPassan position.

3.3.3.5 getPiece()

```
Piece game.Fen.getPiece (
    Pos pos )
```

Return a given piece or null from a given position.

Parameters

<i>pos</i>	The position.
------------	---------------

Returns

The piece at that position.

3.3.3.6 getPlayer()

```
Team game.Fen.getPlayer ( )
```

Return the player currently on the move.

Returns

The current player.

3.3.3.7 getTile()

```
Tile game.Fen.getTile (
    Pos pos )
```

Returns a tile at a given position.

Parameters

<i>pos</i>	The given position.
------------	---------------------

Returns

The tile at the position.

3.3.3.8 getTiles()

```
Tile [][] game.Fen.getTiles ( )
```

Returns a 8x8 array of tiles.

Returns

The array of tiles.

3.3.3.9 setCastling()

```
void game.Fen.setCastling (
    int inx,
    boolean value )
```

Sets the castling currently available.

Parameters

<i>inx</i>	0-WKingside, 1-WQueenside, 2-BKingside, 3-BQueenside.
<i>value</i>	New value of the given index.

3.3.3.10 setEnpassan()

```
void game.Fen.setEnpassan (
    Pos enpassan )
```

Sets a new EnPassan position.

Parameters

<i>enpassan</i>	The new position.
-----------------	-------------------

3.3.3.11 setPlayer()

```
void game.Fen.setPlayer (
    Team player )
```

Sets the current player to a new one.

Parameters

<i>player</i>	New players color.
---------------	--------------------

3.3.3.12 setTile()

```
void game.Fen.setTile (
    Tile tile )
```

Sets a tile into its position in the array.

Parameters

<i>tile</i>	The tile.
-------------	-----------

3.3.3.13 toString()

```
String game.Fen.toString ( )
```

Transforms the object into a readable fen.

Returns

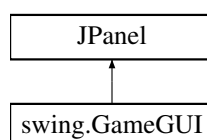
The object in String format.

The documentation for this class was generated from the following file:

- src/game/Fen.java

3.4 swing.GameGUI Class Reference

Inheritance diagram for swing.GameGUI:



Public Member Functions

- [GameGUI](#) ()
- void [setState](#) ([State](#) state)
- [State](#) [getState](#) ()

3.4.1 Detailed Description

The Graphical interface of the chessboard.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 GameGUI()

```
swing.GameGUI.GameGUI ( )
```

Constructs the JPanel with a 9x9 grid and a toolbar.

3.4.3 Member Function Documentation

3.4.3.1 getState()

```
State swing.GameGUI.getState ( )
```

Gets the current state of the gui.

Returns

The current State.

3.4.3.2 setState()

```
void swing.GameGUI.setState (
    State state )
```

Sets the state of the game, GAME if currently playing else MENU

Parameters

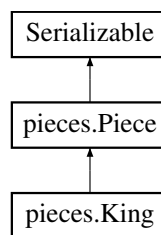
<i>state</i>	Settable state.
--------------	-----------------

The documentation for this class was generated from the following file:

- `src/swing/GameGUI.java`

3.5 pieces.King Class Reference

Inheritance diagram for pieces.King:



Public Member Functions

- **King** ([Team](#) color)
- boolean **validMove** ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Additional Inherited Members

3.5.1 Detailed Description

Classic chess [King](#).

3.5.2 Member Function Documentation

3.5.2.1 validMove()

```
boolean pieces.King.validMove (  
    Pos from,  
    Pos to,  
    Fen fen )
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To positon.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

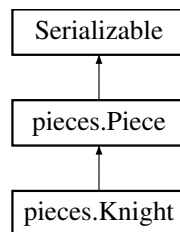
Reimplemented from [pieces.Piece](#).

The documentation for this class was generated from the following file:

- `src/pieces/King.java`

3.6 pieces.Knight Class Reference

Inheritance diagram for pieces.Knight:



Public Member Functions

- **Knight** ([Team](#) color)
- boolean **validMove** ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Additional Inherited Members

3.6.1 Detailed Description

Classic chess [Knight](#).

3.6.2 Member Function Documentation

3.6.2.1 validMove()

```

boolean pieces.Knight.validMove (
    Pos from,
    Pos to,
    Fen fen )
  
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To positon.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

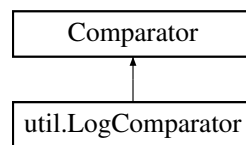
Reimplemented from [pieces.Piece](#).

The documentation for this class was generated from the following file:

- `src/pieces/Knight.java`

3.7 util.LogComparator Class Reference

Inheritance diagram for util.LogComparator:

**Public Member Functions**

- `int compare (Object o1, Object o2)`

3.7.1 Detailed Description

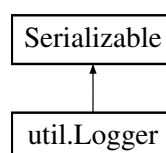
Comparator interface class for sorting Loggers.

The documentation for this class was generated from the following file:

- `src/util/LogComparator.java`

3.8 util.Logger Class Reference

Inheritance diagram for util.Logger:



Public Member Functions

- [Logger](#) ()
- [Logger](#) (String path)
- void [put](#) ([Fen](#) fen)
- HashMap< Integer, [Fen](#) > [getMoves](#) ()
- [Fen](#) [get](#) (int key)
- void [drop](#) ()
- [Fen](#) [getLast](#) ()
- void [save](#) (String name)
- void [configEnd](#) (int inx)
- String [getName](#) ()

3.8.1 Detailed Description

The main class of saving the game, so it has to be serializable.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 [Logger\(\)](#) [1/2]

```
util.Logger.Logger ( )
```

Creates a new logger which has the name of the current time.

3.8.2.2 [Logger\(\)](#) [2/2]

```
util.Logger.Logger (
    String path )
```

Open an already existing save based on its path.

Parameters

<i>path</i>	The path of the directory.
-------------	----------------------------

3.8.3 Member Function Documentation

3.8.3.1 [configEnd\(\)](#)

```
void util.Logger.configEnd (
    int inx )
```

Configures the endstate of a game.

Parameters

<i>inx</i>	0 - Mate, 1 - Stalemate.
------------	--------------------------

3.8.3.2 drop()

```
void util.Logger.drop ( )
```

Deletes the last state from the HashMap, used in reversing a move.

3.8.3.3 get()

```
Fen util.Logger.get (
    int key )
```

Return a value based on its key(index).

Parameters

<i>key</i>	The keyof the value.
------------	----------------------

Returns

The needed state.

3.8.3.4 getLast()

```
Fen util.Logger.getLast ( )
```

Returns the last value put into the HashMap

Returns

The last value.

3.8.3.5 getMoves()

```
HashMap<Integer, Fen> util.Logger.getMoves ( )
```

Gives back the HashMap which stores the moves.

Returns

The HashMap.

3.8.3.6 getName()

```
String util.Logger.getName ( )
```

Returns the path of the [Logger](#).

Returns

The path.

3.8.3.7 put()

```
void util.Logger.put (
    Fen fen )
```

Puts a new move into the logger.

Parameters

<i>fen</i>	The new value.
------------	----------------

3.8.3.8 save()

```
void util.Logger.save (
    String name )
```

Saves the logger into a text file

Parameters

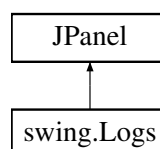
<i>name</i>	The name of the file if blank, random.
-------------	--

The documentation for this class was generated from the following file:

- src/util/Logger.java

3.9 swing.Logs Class Reference

Inheritance diagram for swing.Logs:



Public Member Functions

- [Logs](#) ()
- void [initNames](#) ()
- void [setState](#) ([State](#) state)
- [State](#) [getState](#) ()

3.9.1 Detailed Description

The logs window which contains the saved games.

3.9.2 Constructor & Destructor Documentation

3.9.2.1 Logs()

```
swing.Logs.Logs ( )
```

Constructs the JPanel.

3.9.3 Member Function Documentation

3.9.3.1 getState()

```
State swing.Logs.getState ( )
```

Return the current state.

Returns

The current state.

3.9.3.2 initNames()

```
void swing.Logs.initNames ( )
```

Reads the saved games into an ArrayList.

3.9.3.3 setState()

```
void swing.Logs.setState (
    State state )
```

Sets the state of the window, if running LOG, else MENU.

Parameters

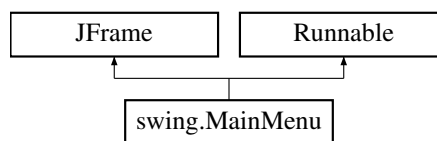
<code>state</code>	The wanted state.
--------------------	-------------------

The documentation for this class was generated from the following file:

- `src/swing/Logs.java`

3.10 swing.MainMenu Class Reference

Inheritance diagram for swing.MainMenu:



Public Member Functions

- [MainMenu](#) ()
- void [run](#) ()

Static Public Member Functions

- static void **main** (String[] args)

3.10.1 Detailed Description

Simple menu with 2 buttons and a title. Runnable so we can navigate back from the submenus.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 MainMenu()

```
swing.MainMenu.MainMenu ( )
```

Constructs the window.

3.10.3 Member Function Documentation

3.10.3.1 run()

```
void swing.MainMenu.run ( )
```

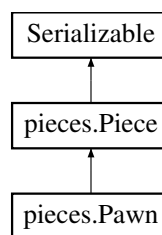
Runnable interface implementation, simply checks which state the application is, and shown the appropriate window.

The documentation for this class was generated from the following file:

- src/swing/MainMenu.java

3.11 pieces.Pawn Class Reference

Inheritance diagram for pieces.Pawn:



Public Member Functions

- **Pawn** ([Team](#) color)
- boolean **validMove** ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Additional Inherited Members

3.11.1 Detailed Description

Classic chess [Pawn](#).

3.11.2 Member Function Documentation

3.11.2.1 validMove()

```
boolean pieces.Pawn.validMove (
    Pos from,
    Pos to,
    Fen fen )
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To positon.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

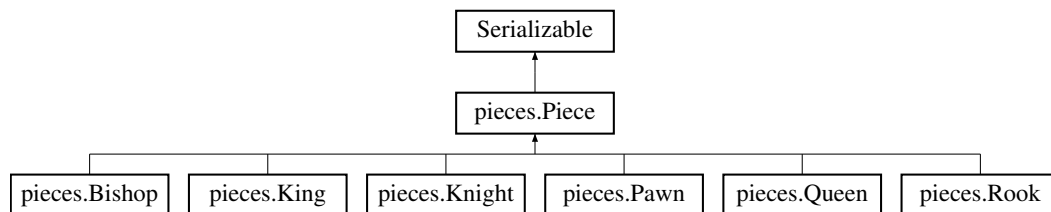
Reimplemented from [pieces.Piece](#).

The documentation for this class was generated from the following file:

- `src/pieces/Pawn.java`

3.12 pieces.Piece Class Reference

Inheritance diagram for pieces.Piece:



Public Member Functions

- boolean [equals](#) (Object o)
- [Team](#) [getColor](#) ()
- [PieceType](#) [getType](#) ()
- abstract boolean [validMove](#) ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Protected Attributes

- [Team](#) [color](#)
- [PieceType](#) [type](#)

3.12.1 Detailed Description

Abstract piece class. Describes a chess piece. Serializable because of io saving.

3.12.2 Member Function Documentation

3.12.2.1 equals()

```
boolean pieces.Piece.equals (
    Object o )
```

Equals override to compare pieces.

Parameters

<i>o</i>	Other piece.
----------	--------------

Returns

True if they are equal.

3.12.2.2 getColor()

```
Team pieces.Piece.getColor ( )
```

Return the color of the piece.

Returns

[Piece](#)'s color.

3.12.2.3 getType()

```
PieceType pieces.Piece.getType ( )
```

Return the type of the piece.

Returns

[Piece](#)'s type.

3.12.2.4 validMove()

```
abstract boolean pieces.Piece.validMove (
    Pos from,
    Pos to,
    Fen fen ) [abstract]
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To positon.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

Reimplemented in [pieces.Rook](#), [pieces.Queen](#), [pieces.Pawn](#), [pieces.Knight](#), [pieces.King](#), and [pieces.Bishop](#).

The documentation for this class was generated from the following file:

- [src/pieces/Piece.java](#)

3.13 util.PieceKey Class Reference

Public Member Functions

- **PieceKey** ([PieceType](#) type, [Team](#) color)
- boolean **equals** (Object o)
- int **hashCode** ()

3.13.1 Detailed Description

Stores a type and a color, giving a unique identifier to a piece. Has equals and hashCode functions.

The documentation for this class was generated from the following file:

- [src/util/PieceKey.java](#)

3.14 util.PieceType Enum Reference

Public Attributes

- **pawn**
- **rook**
- **knight**
- **bishop**
- **king**
- **queen**

3.14.1 Detailed Description

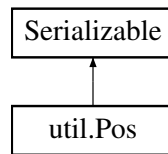
Types of the classic chess pieces.

The documentation for this enum was generated from the following file:

- [src/util/PieceType.java](#)

3.15 util.Pos Class Reference

Inheritance diagram for util.Pos:



Public Member Functions

- **Pos** (int x, int y)
- int **X** ()
- int **Y** ()
- boolean **equals** (Object o)
- int **hashCode** ()

3.15.1 Detailed Description

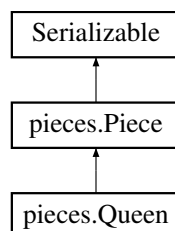
Stores a position, x and y values. Has equals and hashCode functions. Serializable because of io storage.

The documentation for this class was generated from the following file:

- src/util/Pos.java

3.16 pieces.Queen Class Reference

Inheritance diagram for pieces.Queen:



Public Member Functions

- **Queen** ([Team](#) color)
- boolean **validMove** ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Additional Inherited Members

3.16.1 Detailed Description

Classic chess [Queen](#).

3.16.2 Member Function Documentation

3.16.2.1 validMove()

```
boolean pieces.Queen.validMove (
    Pos from,
    Pos to,
    Fen fen )
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To positon.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

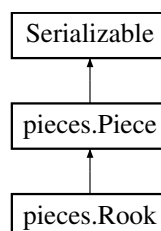
Reimplemented from [pieces.Piece](#).

The documentation for this class was generated from the following file:

- src/pieces/Queen.java

3.17 pieces.Rook Class Reference

Inheritance diagram for pieces.Rook:



Public Member Functions

- **Rook** ([Team](#) color)
- boolean **validMove** ([Pos](#) from, [Pos](#) to, [Fen](#) fen)

Additional Inherited Members

3.17.1 Detailed Description

Classic chess [Rook](#).

3.17.2 Member Function Documentation

3.17.2.1 validMove()

```
boolean pieces.Rook.validMove (
    Pos from,
    Pos to,
    Fen fen )
```

Abstract move validator. Each piece has a different move-set so its implemented differently.

Parameters

<i>from</i>	From position.
<i>to</i>	To position.
<i>fen</i>	The tiles.

Returns

True if the move is in the pieces move-set.

Reimplemented from [pieces.Piece](#).

The documentation for this class was generated from the following file:

- `src/pieces/Rook.java`

3.18 util.State Enum Reference

Public Attributes

- **MENU**
- **GAME**
- **LOG**

3.18.1 Detailed Description

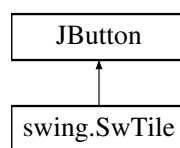
The states of the Application.

The documentation for this enum was generated from the following file:

- `src/util/State.java`

3.19 swing.SwTile Class Reference

Inheritance diagram for swing.SwTile:



Public Member Functions

- `Pos getPos ()`
- `void setPos (Pos pos)`

3.19.1 Detailed Description

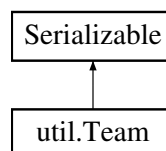
Class that extends JButton with the little extra of storing simple coordinates.

The documentation for this class was generated from the following file:

- `src/swing/SwTile.java`

3.20 util.Team Enum Reference

Inheritance diagram for util.Team:



Public Attributes

- `white`
- `black`

3.20.1 Detailed Description

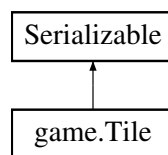
Colors in chess.

The documentation for this enum was generated from the following file:

- `src/util/Team.java`

3.21 game.Tile Class Reference

Inheritance diagram for game.Tile:



Public Member Functions

- [Tile](#) (int x, int y)
- [Tile](#) ([Pos](#) pos)
- [Tile](#) ([Piece](#) piece, [Pos](#) pos)
- [Piece](#) [getPiece](#) ()
- [Pos](#) [getPos](#) ()
- void [setPos](#) ([Pos](#) pos)

3.21.1 Detailed Description

Describes a single tile, its position and what piece is on it. Serializable because of io storage.

3.21.2 Constructor & Destructor Documentation

3.21.2.1 [Tile\(\)](#) [1/3]

```

game.Tile.Tile (
    int x,
    int y )
  
```

Constructs a tile with the given x, y values.

Parameters

<i>x</i>	x value.
<i>y</i>	y value.

3.21.2.2 Tile() [2/3]

```
game.Tile.Tile (
    Pos pos )
```

Constructs a tile with a given position.

Parameters

<i>pos</i>	The given position.
------------	---------------------

3.21.2.3 Tile() [3/3]

```
game.Tile.Tile (
    Piece piece,
    Pos pos )
```

Constructs a tile with a given position and a piece.

Parameters

<i>piece</i>	The piece.
<i>pos</i>	The position.

3.21.3 Member Function Documentation

3.21.3.1 getPiece()

```
Piece game.Tile.getPiece ( )
```

Returns the piece on the tile.

Returns

The piece on the tile.

3.21.3.2 `getPos()`

```
Pos game.Tile.getPos ( )
```

Returns the position of the tile.

Returns

`Tile`'s position.

3.21.3.3 `setPos()`

```
void game.Tile.setPos (
    Pos pos )
```

Sets the tiles position.

Parameters

<code>pos</code>	Wanted position.
------------------	------------------

The documentation for this class was generated from the following file:

- `src/game/Tile.java`

Index

- addTurn
 - game.Fen, [11](#)
- Board
 - game.Board, [6](#), [7](#)
- castling
 - game.Board, [7](#)
- configEnd
 - util.Logger, [19](#)
- drop
 - util.Logger, [20](#)
- equals
 - pieces.Piece, [25](#)
- executeMove
 - game.Board, [7](#)
- Fen
 - game.Fen, [10](#)
- game.Board, [6](#)
 - Board, [6](#), [7](#)
 - castling, [7](#)
 - executeMove, [7](#)
 - getFen, [8](#)
 - inCheck, [8](#)
 - isMate, [8](#)
 - isStalemate, [8](#)
 - validMove, [9](#)
- game.Fen, [10](#)
 - addTurn, [11](#)
 - Fen, [10](#)
 - getCastling, [11](#)
 - getCastlingAt, [11](#)
 - getEnpassan, [11](#)
 - getPiece, [12](#)
 - getPlayer, [12](#)
 - getTile, [12](#)
 - getTiles, [13](#)
 - setCastling, [13](#)
 - setEnpassan, [13](#)
 - setPlayer, [13](#)
 - setTile, [14](#)
 - toString, [14](#)
- game.Tile, [32](#)
 - getPiece, [33](#)
 - getPos, [33](#)
 - setPos, [34](#)
 - Tile, [32](#), [33](#)
- GameGUI
 - swing.GameGUI, [15](#)
- get
 - util.Logger, [20](#)
- getCastling
 - game.Fen, [11](#)
- getCastlingAt
 - game.Fen, [11](#)
- getColor
 - pieces.Piece, [26](#)
- getEnpassan
 - game.Fen, [11](#)
- getFen
 - game.Board, [8](#)
- getLast
 - util.Logger, [20](#)
- getMoves
 - util.Logger, [20](#)
- getName
 - util.Logger, [20](#)
- getPiece
 - game.Fen, [12](#)
 - game.Tile, [33](#)
- getPlayer
 - game.Fen, [12](#)
- getPos
 - game.Tile, [33](#)
- getState
 - swing.GameGUI, [15](#)
 - swing.Logs, [22](#)
- getTile
 - game.Fen, [12](#)
- getTiles
 - game.Fen, [13](#)
- getType
 - pieces.Piece, [26](#)
- inCheck
 - game.Board, [8](#)
- initNames
 - swing.Logs, [22](#)
- isMate
 - game.Board, [8](#)
- isStalemate
 - game.Board, [8](#)
- Logger
 - util.Logger, [19](#)
- Logs
 - swing.Logs, [22](#)

- MainMenu
 - swing.MainMenu, 23
- pieces.Bishop, 5
 - validMove, 5
- pieces.King, 16
 - validMove, 16
- pieces.Knight, 17
 - validMove, 17
- pieces.Pawn, 24
 - validMove, 24
- pieces.Piece, 25
 - equals, 25
 - getColor, 26
 - getType, 26
 - validMove, 26
- pieces.Queen, 28
 - validMove, 29
- pieces.Rook, 29
 - validMove, 30
- put
 - util.Logger, 21
- run
 - swing.MainMenu, 23
- save
 - util.Logger, 21
- setCastling
 - game.Fen, 13
- setEnpassan
 - game.Fen, 13
- setPlayer
 - game.Fen, 13
- setPos
 - game.Tile, 34
- setState
 - swing.GameGUI, 15
 - swing.Logs, 22
- setTile
 - game.Fen, 14
- swing.GameGUI, 14
 - GameGUI, 15
 - getState, 15
 - setState, 15
- swing.Logs, 21
 - getState, 22
 - initNames, 22
 - Logs, 22
 - setState, 22
- swing.MainMenu, 23
 - MainMenu, 23
 - run, 23
- swing.SwTile, 31
- Tile
 - game.Tile, 32, 33
- toString
 - game.Fen, 14
- util.LogComparator, 18
- util.Logger, 18
 - configEnd, 19
 - drop, 20
 - get, 20
 - getLast, 20
 - getMoves, 20
 - getName, 20
 - Logger, 19
 - put, 21
 - save, 21
- util.PieceKey, 27
- util.PieceType, 27
- util.Pos, 28
- util.State, 30
- util.Team, 31
- validMove
 - game.Board, 9
 - pieces.Bishop, 5
 - pieces.King, 16
 - pieces.Knight, 17
 - pieces.Pawn, 24
 - pieces.Piece, 26
 - pieces.Queen, 29
 - pieces.Rook, 30