Mancala Team 12

Generated by Doxygen 1.8.13

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 AI Class Reference

Public Member Functions

- AI (int _maxDepth, boolean _player)
- AI (boolean _player, int _timer)
- double eval (Board node, boolean playerOneOrTwo, int _maxDepth)
- double miniMax (Board board, int depthToGo, boolean playerOneOrTwo, double alpha, double beta)
- int getNumNodes ()
- int getBestMove ()

3.1.1 Constructor & Destructor Documentation

3.1.1.1 AI() [1/2]

```
AI.AI (
             int _maxDepth,
             boolean _player )
```

Initializes an AI with a maxDepth

Parameters

_maxDepth	The number of levels that the minimax function will go through.

3.1.1.2 AI() [2/2]

```
AI.AI (
            boolean _player,
            int _timer )
```

Initializes an AI with a time limit. Will calculate the max depth an AI will go.

Parameters

_player	Which player the AI should be.
_timer	How long the timer is in milliseconds

3.1.2 Member Function Documentation

3.1.2.1 eval()

```
double AI.eval (
            Board node,
            boolean playerOneOrTwo,
            int _maxDepth )
```

Evaluates a board "node" for player one or two Returns the best utility for the board. Used in the calculate maxDepth

Parameters

node	
playerOneOrTwo	
_maxDepth	the maximum depth.

Returns

a utility function for the board.

3.1.2.2 getBestMove()

```
int AI.getBestMove ( )
```

Getter method for the best move. Used in conjunction with the eval function to get the best move in the Client.

Returns

the best move with certain depth.

3.1.2.3 getNumNodes()

```
int AI.getNumNodes ( )
```

Getter method for number of nodes in minimax tree. Used mainly for debuging purposes

Returns

the size of the tree.

3.1.2.4 miniMax()

```
double AI.miniMax (
             Board board,
             int depthToGo,
             boolean playerOneOrTwo,
             double alpha,
             double beta )
```

The minimax function for use in the eval function. Implements alpha beta pruning and pie move recognition.

Parameters

board	The board you want to evaluate
depthToGo	The depth left before it stops.
playerOneOrTwo	The player that it's calculating at that level
alpha	The alpha cutoff
beta	The beta cutoff

Returns

a utility function for the children

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

- Mancala/src/AI.java

3.2 AIClient Class Reference

Public Member Functions

- AIClient () throws UnknownHostException, IOException
- void play () throws IOException

Static Public Member Functions

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

3.2.1 Constructor & Destructor Documentation

3.2.1.1 AIClient()

```
AIClient.AIClient ( ) throws UnknownHostException, IOException
```

Constructs clients by connecting to server and laying out the start GUI Start GUI tells whether it is person - person or person - AI game. This determines how the client runs

Exceptions

IOException	
UnknownHostException	
Exception	

3.2.2 Member Function Documentation

3.2.2.1 play()

```
void AIClient.play ( ) throws IOException
```

Main thread will listen for messages from the server

Exceptions

IOException	If it fails in sending/receiving
InterruptedException	If the timer stops
Exception	other exceptions

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

- Mancala/src/AIClient.java

3.3 AITest Class Reference

Public Member Functions

- void **bestMove1** ()
- void **bestMove2** ()
- void **bestMove3** ()

- void **bestMove4** ()

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

- Mancala/src/AITest.java

3.4 Board Class Reference

Public Member Functions

- Board (int _numHouses, int _numSeeds, boolean _randomSeeds)
- **Board** (int _numHouses, int _numSeeds, boolean _randomSeeds, int[ ] randomSeeds)
- int [ ] getRandomArray ()
- Board (int [ ][ ] _board, int _score1, int _score2, int _numHouses)
- String toString ()
- ArrayList< Board > getChildren (boolean playerOneOrTwo)
- Board cloneChild (Board toCopy)
- Board copy (Board toCopy)
- boolean extraTurn (int _row, int _column, boolean _pieRule)
- boolean movePiece (int _row, int _column, boolean _pieRule)
- ArrayList< Integer > validMoves (boolean playerOneOrTwo)
- boolean validMove (int row, int column)
- boolean checkEndState ()
- String **outcome** ()
- int [ ][ ] getBoardArray ()
- int getPlayerOneStore ()
- int getPlayerTwoStore ()
- void setPlayerOneStore (int _storePlayer1)
- void setPlayerTwoStore (int _storePlayer2)
- void initializeIndexMappings (int houses)
- void calculatePitsOnSides ()
- int utility (boolean playerOneOrTwo)

Public Attributes

- int [ ][ ] **board**
- int **storePlayer1**
- int **moveCounter** = 0
- Board **parent** = null
- int **previous** = -999
- int **pitsOnSide1** = 0
- int **pitsOnSide2** = 0
- boolean **pieMoveConsidered**

3.4.1 Constructor & Destructor Documentation

3.4.1.1 Board() [1/2]

```
Board.Board (
                int _numHouses,
                int _numSeeds,
                boolean _randomSeeds )
```

Initializes a board for use in the client/server.

Parameters

_numHouses	The number of houses. Can be from 4-9.
_numSeeds	Can be from 4-9.
_randomSeeds	True if you want to randomly distribute the seeds.

3.4.1.2 Board() [2/2]

```
Board.Board (
             int _board[ ][ ],
             int _score1,
             int _score2,
             int _numHouses )
```

Secondary Constructor takes in a 2d array of ints that represents a board and the store of each player. Doesn't check for board validity or end state. Used in the clone and test functions.

Parameters

_board	2d array of ints of size 2xnumHouses. Throws Invalid board error if wrong dimensions.
_score1	store of player 1.
_score2	store of player 2.
_numHouses	the number of houses.

3.4.2 Member Function Documentation

3.4.2.1 calculatePitsOnSides()

```
void Board.calculatePitsOnSides ( )
```

Calculates the total number of pits on both player's houses Used in the utility function.

3.4.2.2 checkEndState()

```
boolean Board.checkEndState ( )
```

Check if the game is over. If one side is empty, the game is over and all of the stones on the other side go to the other player's store. This method will move them there and return true or false.

Returns

true if the game is over else false.

3.4.2.3 cloneChild()

```
Board Board.cloneChild (
             Board toCopy )
```

Clones a parent and sets the child's parent to toCopy

Parameters

toCopy	

Returns

3.4.2.4 copy()

```
Board Board.copy (
             Board toCopy )
```

Copies the board.

Parameters

toCopy	board you want to copy.

Returns

the copied board

3.4.2.5 extraTurn()

```
boolean Board.extraTurn (
             int _row,
             int _column,
             boolean _pieRule )
```

Determines if a move will get an extra turn. Used in the runningGame method

Parameters

_row	The row that the player picks. 0 is assumed to be for player 1 and 1 for player 2. -1 if using pieRule.
_column	The column that the player picks and assumes player picks 0-5. -1 if using pieRule.
_pieRule	true if you want to use the pieRule else false. Make sure _row=_column=-1

Returns

3.4.2.6 getBoardArray()

```
int [][] Board.getBoardArray ( )
```

Getter method for the 2d array.

Returns

2d array of ints representing the board.

3.4.2.7 getChildren()

```
ArrayList<Board> Board.getChildren (
            boolean playerOneOrTwo )
```

Returns an array of all the children of a certain board. The children are the possible boards that can be made in one player's turn including the pie move. If the child has another move, then we include the next move. Used in the miniMax function.

Parameters

playerOneOrTwo	

Returns

3.4.2.8 getPlayerOneStore()

```
int Board.getPlayerOneStore ( )
```

Getter method for the store of player 1.

Returns

Returns an int of player one's store.

3.4.2.9 getPlayerTwoStore()

```
int Board.getPlayerTwoStore ( )
```

Getter method for the store of player 2.

Returns

Returns an int of player two's store.

3.4.2.10 getRandomArray()

```
int [] Board.getRandomArray ( )
```

Returns the randomly created array for use in DataServer.

Returns

Randomly distributed array.

3.4.2.11 initializeIndexMappings()

```
void Board.initializeIndexMappings (
            int houses )
```

Initializes the indexToRow and indexToCol arrays for use in constructors.

Parameters

houses	the number of houses you want.

3.4.2.12 movePiece()

```
boolean Board.movePiece (
            int _row,
            int _column,
            boolean _pieRule )
```

Moves pieces given a row and column and returns a boolean if the player gets to go again. Doesn't check for end state. Doesn't have a check for player turn, but will throw an exception if the player makes an invalid move.

Parameters

_row	The row that the player picks. 0 is assumed to be for player 1 and 1 for player 2. -1 if using pieRule.
_column	The column that the player picks and assumes player picks 0-5. -1 if using pieRule.
_pieRule	true if you want to use the pieRule else false. Make sure _row=_column=-1

Returns

true if the player gets to go again(when he/she lands on mancala) else false.

3.4.2.13 setPlayerOneStore()

```
void Board.setPlayerOneStore (
            int _storePlayer1 )
```

Setter method for player 1's store

Parameters

_storePlayer1	the number you want to set it to

3.4.2.14 setPlayerTwoStore()

```
void Board.setPlayerTwoStore (
            int _storePlayer2 )
```

Setter method for player 2's store

Parameters

_storePlayer1	the number you want to set it to

3.4.2.15 toString()

```
String Board.toString ( )
```

Creates a string representation of the board for debugging purposes

3.4.2.16 utility()

```
int Board.utility (
            boolean playerOneOrTwo )
```

Evaluates a board and gives it a utility value It prioritizes scoring first and the number of pits in the player's houses second

Returns

an integer representing the utility of that board.

3.4.2.17 validMove()

```
boolean Board.validMove (
            int row,
            int column )
```

Determine if a move is valid.

Parameters

row	The row of the move you want to make.
column	The column of the move you want to make.

Returns

true if it's valid and false if it is.

3.4.2.18 validMoves()

```
ArrayList<Integer> Board.validMoves (
            boolean playerOneOrTwo )
```

Gives the valid move choices for a player. Used in the getChildren method.

Parameters

playerOneOrTwo	true if playerOne and false for playerTwo

Returns

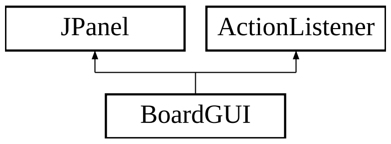
Arraylist of integers that represent the columns that the player can choose.

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

- Mancala/src/Board.java

3.5 BoardGUI Class Reference

Inheritance diagram for BoardGUI:



Public Member Functions

- void setTurn (Boolean _inBool)
- void update (Board _inBoard)
- void **actionPerformed** (ActionEvent e)
- boolean getPiRule ()
- boolean falsePiRule ()

Public Attributes

- boolean **player1** = true
- boolean **timeUp** = false
- boolean **endGame** = false
- boolean **piRule** = false
- boolean **secondTurnHappened** = false

Static Public Attributes

- static final java.awt.Color **BURLY_WOOD** = new Color(222, 184, 135)
- static final Color **BlanchedAlmond** = new Color(255, 235, 205)
- static int **newBoard** [ ][ ]

3.5.1 Member Function Documentation

3.5.1.1 falsePiRule()

```
boolean BoardGUI.falsePiRule ( )
```

Turns the pie rule off.

Returns

Returns false after the pie rule has be turned off.

3.5.1.2 getPiRule()

```
boolean BoardGUI.getPiRule ( )
```

Returns if the pie rule has been used.

Returns

3.5.1.3 setTurn()

```
void BoardGUI.setTurn (
             Boolean _inBool )
```

Sets current players turn

Parameters

_inBool,sets	player

3.5.1.4 update()

```
void BoardGUI.update (
             Board _inBoard )
```

updates the Board BUI

Parameters

_inBoard,board	used to paint

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

- Mancala/src/BoardGUI.java

3.6 BoardTest Class Reference

Public Member Functions

- void **testSecondaryConstructor** ()
- void **testMovePiece** ()

- void **testMovePiece2** ()
- void **testMovePiece3** ()
- void **testMovePiece4** ()
- void **testMovePiece5** ()
- void **testMovePiece6** ()
- void **testMovePiece7** ()
- void **testMovePiece8** ()
- void **testMovePiece9** ()
- void **testValidMoves** ()
- void **testValidMoves2** ()
- void **testCheckEndState** ()
- void **testCheckEndState2** ()

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

- Mancala/src/BoardTest.java

3.7 ClientFunctions Class Reference

Public Member Functions

- ClientFunctions ()
- ClientFunctions (String[ ] info)
- void launchStartGUI ()
- boolean init (String response)
- void sendReady ()
- String **createMoves** (boolean isPlayerOne)
- void updateBoard (String response, boolean whoMoved, boolean pieMove)
- void endGame (String winOrLose)
- void recieveBoardInfo (String boardInfo)
- void recieveBegin ()
- void toggleTurn ()
- void receiveWelcome (String welcome)
- void myTurn ()
- void endMyTurn ()

Public Attributes

- GameManager **gameManager**
- boolean **isFirst**

3.7.1 Constructor & Destructor Documentation

3.7.1.1 ClientFunctions() [1/2]

```
ClientFunctions.ClientFunctions ( )
```

Client functions run differently per human or AI

3.7.1.2 ClientFunctions() [2/2]

```
ClientFunctions.ClientFunctions (
              String [ ] info )
```

Takes in an info string and creates an AI based on the information given.

Parameters

info	The info string.

3.7.2 Member Function Documentation

3.7.2.1 endGame()

```
void ClientFunctions.endGame (
             String winOrLose )
```

Used to end the game and update the text

Parameters

winOrLose	If the person won, lost or drew.

3.7.2.2 endMyTurn()

```
void ClientFunctions.endMyTurn ( )
```

Updates the bottom text

3.7.2.3 init()

```
boolean ClientFunctions.init (
             String response )
```

Initializes a gameManager/other things you might need Delete if not needed.

3.7.2.4 launchStartGUI()

```
void ClientFunctions.launchStartGUI ( )
```

Launches the game manager's GUI

3.7.2.5 myTurn()

```
void ClientFunctions.myTurn ( )
```

Update the turn indicator

3.7.2.6 receiveWelcome()

```
void ClientFunctions.receiveWelcome (
             String welcome )
```

Start the game once it's received a welcome and to update the bottom text

Parameters

welcome	Should just be "welcome"

3.7.2.7 recieveBegin()

```
void ClientFunctions.recieveBegin ( )
```

Acknowledges that the game has begun. calls sendMove if it's player 1.

3.7.2.8 recieveBoardInfo()

```
void ClientFunctions.recieveBoardInfo (
             String boardInfo )
```

Takes in the board info and sets the correct parameters

Parameters

boardInfo	INFO <int holes="" per="" side>=""><int seeds="" per="" side>=""><long int="" time="" for="" timer>=""><F|S><S|R hole_config>

3.7.2.9 sendReady()

```
void ClientFunctions.sendReady ( )
```

Send to server that it's ready

3.7.2.10 toggleTurn()

```
void ClientFunctions.toggleTurn ( )
```

Toggle the turn to display who's turn it is.

3.7.2.11 updateBoard()

```
void ClientFunctions.updateBoard (
             String response,
             boolean whoMoved,
             boolean pieMove )
```

Updates the board on the screen after receiving a response from the server

Parameters

response	The response string from server
whoMoved	The last person to move
pieMove	Whether or not to use the pie move.

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

- Mancala/src/ClientFunctions.java

3.8 DataClient Class Reference

Public Member Functions

- DataClient () throws UnknownHostException, IOException
- void play () throws IOException

Static Public Member Functions

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

3.8.1 Constructor & Destructor Documentation

3.8.1.1 DataClient()

```
DataClient.DataClient ( ) throws UnknownHostException, IOException
```

Constructs clients by connecting to server and laying out the start GUI Start GUI tells whether it is person - person or person - AI game. This determines how the client runs

Exceptions

IOException	
UnknownHostException	
Exception	

3.8.2 Member Function Documentation

3.8.2.1 play()

```
void DataClient.play ( ) throws IOException
```

Main thread will listen for messages from the server

Exceptions

IOException	
Exception	

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

- Mancala/src/DataClient.java

3.9 DataServer Class Reference

Static Public Member Functions

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

3.9.1 Detailed Description

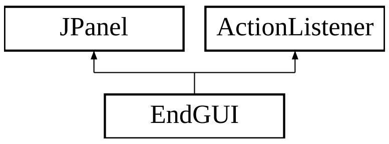
A TCP server that runs on port 9090. When a client connects, it sends the client the current date and time, then closes the connection with that client. Arguably just about the simplest server you can write.

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

- Mancala/src/DataServer.java

3.10 EndGUI Class Reference

Inheritance diagram for EndGUI:



Public Member Functions

- EndGUI (int _player1, int _player2)
- void paint (Graphics g)
- void **actionPerformed** (ActionEvent e)

3.10.1 Constructor & Destructor Documentation

3.10.1.1 EndGUI()

```
EndGUI.EndGUI (
              int _player1,
              int _player2 )
```

Builds the GUI for the ending screen

Parameters

player1	the score of player 1
player2	the score of player 2

3.10.2 Member Function Documentation

3.10.2.1 paint()

```
void EndGUI.paint (
              Graphics g )
```

(non-Javadoc)

See also

javax.swing.JComponent::paint(java.awt.Graphics)

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

- Mancala/src/EndGUI.java

3.11 GameManager Class Reference

Public Member Functions

- GameManager ()
- void launchStartGUI ()
- void initGame (int _holes, boolean _randomSeeds, boolean _timer, int _timerLength)
- void runningGame (GameScreen game, int _numHoles, Boolean randomSeeds)
- void clientLaunchStartGUI () throws Exception
- void clientInitGame (int holes, Boolean randomSeeds, Boolean timer, int timerLength, boolean isAI, int num←Seeds)
- String clientRunningGame (GameScreen game, int numHoles, boolean _isPlayerOne)
- void **clientEndGame** ()
- int getMove ()
- void **setRandomArray** (int[ ] _inArray)

Static Public Member Functions

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

Public Attributes

- GameScreen **game** = null

3.11.1 Detailed Description

Game Manager coordinates screen launches

3.11.2 Constructor & Destructor Documentation

3.11.2.1 GameManager()

```
GameManager.GameManager ( )
```

Empty constructor Game Manager

3.11.3 Member Function Documentation

3.11.3.1 clientInitGame()

```
void GameManager.clientInitGame (
            int holes,
            Boolean randomSeeds,
            Boolean timer,
            int timerLength,
            boolean isAI,
            int numSeeds )
```

Initializes the game for use in the client move.

Parameters

_holes	The number of holes in the board.
_randomSeeds	True to randomly distribute the seeds else false.
_timer	True if you want to have a timer.
_timerLength	The length of time.

3.11.3.2 clientLaunchStartGUI()

```
void GameManager.clientLaunchStartGUI ( ) throws Exception
```

Exceptions

IOException	
UnknownHostException	

3.11.3.3 clientRunningGame()

```
String GameManager.clientRunningGame (
             GameScreen game,
             int numHoles,
             boolean _isPlayerOne )
```

Used for running the game in createMoves.

Parameters

game	The gamescreen of the client
numHoles	The number of holes in the board
_isPlayerOne	The client's player.

Returns

3.11.3.4 getMove()

```
int GameManager.getMove ( )
```

Returns

3.11.3.5 initGame()

```
void GameManager.initGame (
             int _holes,
             boolean _randomSeeds,
             boolean _timer,
             int _timerLength )
```

Initializes the GameScreen and Board

Parameters

_holes	The number of holes in the board.
_randomSeeds	True to randomly distribute the seeds else false.
_timer	True if you want to have a timer.
_timerLength	The length of time.

3.11.3.6 launchStartGUI()

```
void GameManager.launchStartGUI ( )
```

creates start screen

3.11.3.7 runningGame()

```
void GameManager.runningGame (
             GameScreen game,
             int _numHoles,
             Boolean randomSeeds )
```

watches the running game and takes care of clicks accordingly

Parameters

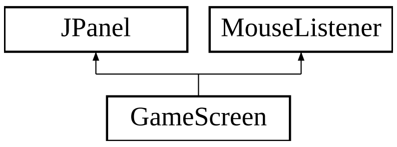
game,the	current GameScreen
board,the	current board
randomSeeds	DOESN"t GET USED

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

- Mancala/src/GameManager.java

3.12 GameScreen Class Reference

Inheritance diagram for GameScreen:



Public Member Functions

- void resetVariables ()
- void createAndShowGUI (Board _inBoard, int numHoles)
- int getRow ()
- int getColumn ()
- Boolean getExtraTurn ()
- void setExtraTurn (boolean _inExtra)
- void setClientGameOver (Boolean endGame)
- void mouseClicked (MouseEvent e)
- void **mousePressed** (MouseEvent e)
- void **mouseReleased** (MouseEvent e)
- void **mouseEntered** (MouseEvent e)
- void **mouseExited** (MouseEvent e)

Public Attributes

- boolean **timeUp** = false
- JFrame **frame** = new JFrame("MANCALA")
- boolean **endGame** = false
- boolean **replay**
- JLabel **messageLabel** = new JLabel("MESSAGE LABEL",SwingConstants.CENTER)

Static Public Attributes

- static boolean **didClick** = false
- static boolean **player1** = true
- static int **newBoard** [ ][ ]

3.12.1 Detailed Description

Game Screen takes care of click events and managing GUI's

3.12.2 Member Function Documentation

3.12.2.1 createAndShowGUI()

```
void GameScreen.createAndShowGUI (
            Board _inBoard,
            int numHoles )
```

Create the GUI and show it.

Parameters

Board	b2 is the board that is being drawn

3.12.2.2 getColumn()

```
int GameScreen.getColumn ( )
```

determines which column was clicked

Returns

column of the click

3.12.2.3 getExtraTurn()

```
Boolean GameScreen.getExtraTurn ( )
```

Determines if there is an extra turn.

Returns

3.12.2.4 getRow()

```
int GameScreen.getRow ( )
```

determines which row was clicked

Returns

row of the click

3.12.2.5 mouseClicked()

```
void GameScreen.mouseClicked (
             MouseEvent e )
```

Gets the pot number that was clicked so that we can distribute seeds as needed Also checks if the player gets an extra turn (non-Javadoc)

See also

java.awt.event.MouseListener::mouseClicked(java.awt.event.MouseEvent)

3.12.2.6 resetVariables()

```
void GameScreen.resetVariables ( )
```

Resets the variables for redrawing and ending the game

3.12.2.7 setClientGameOver()

```
void GameScreen.setClientGameOver (
             Boolean endGame )
```

Stops the game from the client.

Parameters

endGame	Ends the game if true

3.12.2.8 setExtraTurn()

```
void GameScreen.setExtraTurn (
            boolean _inExtra )
```

Sets if there is an extra turn

Parameters

_inExtra	true if there is an extra turn else false

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

- Mancala/src/GameScreen.java

3.13 OptionGUI Class Reference

Public Member Functions

- String getINFO ()
- OptionGUI ()
- void Launch ()

Static Public Attributes

- static JTextField **inputH**
- static JTextField **inputS**
- static JTextField **inputT**

3.13.1 Constructor & Destructor Documentation

3.13.1.1 OptionGUI()

```
OptionGUI.OptionGUI ( )
```

default constructor that calls the gui.

3.13.2 Member Function Documentation

3.13.2.1 getINFO()

```
String OptionGUI.getINFO ( )
```

Getter method for the information

Returns

3.13.2.2 Launch()

```
void OptionGUI.Launch ( )
```

Set the launch button to visible once all the fields are set.

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

- Mancala/src/OptionGUI.java

3.14 StartGUI Class Reference

Public Member Functions

- String getMode ()
- StartGUI ()

Static Public Attributes

- static JTextField **input**

3.14.1 Constructor & Destructor Documentation

3.14.1.1 StartGUI()

```
StartGUI.StartGUI ( )
```

Create the start screen

3.14.2 Member Function Documentation

3.14.2.1 getMode()

```
String StartGUI.getMode ( )
```

Get the mode from the start GUI

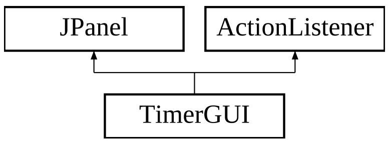
Returns

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

- Mancala/src/StartGUI.java

3.15 TimerGUI Class Reference

Inheritance diagram for TimerGUI:



Public Member Functions

- void update ()
- void **serverUpdate** ()
- TimerGUI ()
- void **pause** (boolean pauseToggle)
- void **wait** (boolean waitToggle)
- void paint (Graphics g)
- void **actionPerformed** (ActionEvent e)

Public Attributes

- boolean **gameOver** = false

3.15.1 Constructor & Destructor Documentation

3.15.1.1 TimerGUI()

```
TimerGUI.TimerGUI ( )
```

Constructor sets up the timer

3.15.2 Member Function Documentation

3.15.2.1 paint()

```
void TimerGUI.paint (
             Graphics g )
```

Painting the timer (non-Javadoc)

See also

javax.swing.JComponent::paint(java.awt.Graphics)

3.15.2.2 update()

```
void TimerGUI.update ( )
```

update() is called if a player takes their turn before time is up This sets up the timer for the next player

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

- Mancala/src/TimerGUI.java

Index