Technical Report – Socket based TicTacToe Game

Group members: Andromeda Weir, Areli Lastname, Ben Lastname

Class Overview:

The code base is split across 4 python files, each of which contains at least one class, listed below:

* Main.py
* TicTacToe.py
  + GameEndException
  + TicTacToe
* TicTacToeGUI.py
  + Player
  + TicTacToeBoard
* Connections.py
  + formalConnectionInterface
  + TicTacToeServer
  + TicTacToeClient

TicTacToe class methods:

* Constructor, optional parameters length and width
  + Creates a new instance of the class, parameters length and width are used to determine the dimensions of the TicTacToe board, which can be made an arbitrary size. Defaults to 3 by 3 board
* setConnection, mandatory parameter Connection must implement formalConnectionInterface
  + takes a reference to the established connection from one player to another to allow for the game object to make calls to network functions
* getBoardState:
  + returns a 2 dimensional array representing the state of the game at time of call
* updateBoardState, mandatory parameter newBoardState must be a 2 dimensional array of ints
  + called by a connection object to pass received information to the game, synchronizing the state between both players
  + This function also checks to see if the new board state has a game ending Condition, and throws GameEndException if it does
* getBoardSize:
  + returns an integer value representing the dimension of the board, useful only in non-standard dimensions
* takeTurn, mandatory parameters player, posX, posY:
  + receives the int value of the player making a move, and the desired position that the player wishes to play at, throws ValueError if the move is illegal for any reason (not player’s turn, space already occupied)
* checkWin, mandatory parameters posX, posY, optional parameter checkTurn:
  + checks to see if a move made at a given position results in a win condition
  + if no parameter checkTurn is passed, function checks for win conditions on behalf of the player whose turn it currently is
  + throws GameEndException when a Win or Tie condition is found
* resetGame:
  + resets the state of the game to allow for a new game to be played

formalConnectionInterface class:

* Does not provide any methods of its own, only requires that its subclasses impliment method sendNewState

TicTacToeServer class methods:

* Constructor, parameter game must be TicTacToe object
  + creates, binds, and listens on a socket
  + calls setConnection(self) on the TicTacToe object it is passed, allowing the two objects to call eachother’s public methods
* sendNewState, mandatory parameter boardState must be a 2d list of ints:
  + translates the boardState into a byteArray object and sends it to the other player
* start:
  + accepts a new connection to the server, and starts a thread that begins execution on method handle\_client
* handle\_client, mandatory parameter clientConnection
  + listens for new data to be received, and calls TicTacToe.updateBoardState() with received data when the other player sends move data

TicTacToeClient class methods:

* constructor, mandatory parameters hostIP, portNum, game:
  + Initializes a connection to a server at a given host IP and port number, passes a reference to itself to the game once the connection is established, and creates a new thread beginning execution at recieveData
* sendNewState, mandatory parameter boardState must be a 2d array of ints:
  + translates boardState to bytearray and sends to other player
* recieveData:
  + upon reciept of data from other player, updates the board state to synchronize the state between players

TicTacToeBoard methods:

- Doesn’t have to be an exhaustive list, just the ones that are relevant to understanding how the class works and how they all fit together