## **Client.Client Class Reference**

## **Public Member Functions**

def \_\_init\_\_ (self)

setup the client More...

def createGame (self, username, password)

inform the server we wish to create a game More...

def test (self)

test function More...

def poll (self)

poll function More...

def join (self, address, username, password)

ask to join a specific game with username and password More...

def listGames (self)

Accessor: returns a list of games client has heard about. More...

def addToQueue (self)

Adds data to the queue. More...

## Static Public Attributes

int BROADCAST\_PORT = 44470

int TRANSMIT\_PORT = 44469

string BROADCAST\_ADDRESS = "255.255.255.255"

string BOARD\_FILE = "text/DublinBoard.txt"

### **Private Member Functions**

def \_transmit (self, transmit\_port)

to be run in a thread and handle outgoing messages to specific server More...

def \_newGame (self, data)

handles a GAME message from the server by adding it to the list of open games self the object pointer data the data TODO: wtf is data More...

def \_gameStart (self, data)

handles a START message from server by creating the board and passing it to GUI self the object pointer data the data TODO: wtf is data More...

def \_newTurn (self, data)

handles a TURN message from the server by telling GUI who's turn has begun More...

def \_hasQuit (self, data)

Calls the board.RemovePlayer() method if a player quits the game. More...

def \_bought (self, data)

update the owner of some space in board to be player with given id More...

def \_paid (self, data)

```
update players balances as they change, 1 or more depending on transaction More...

def __jailed (self, data)
    some player got sent to jail so change their jail status More...

def __sentchat (self, data)
    send a message from the server to the textbox display More...

def __drewCard (self, data)
    you drew a card so tell the GUI about it and check if you're on bail More...
```

### **Private Attributes**

```
_connection_queue
_socket
_transmitter
_open_games
_board
_local_player
```

## Constructor & Destructor Documentation

```
    __init__()

def Client.Client.__init__ ( self )

setup the client
```

### Member Function Documentation

```
    ▶ _bought()
    def Client.Client._bought ( self, data )
    update the owner of some space in board to be player with given id
    Parameters
    self the object pointer
    dict data the data stream from client & server
```

```
_drewCard()
```

```
def Client.Client._drewCard(self,
data
)

you drew a card so tell the GUI about it and check if you're on bail

Parameters
self the object pointer
dict data the data stream from client & server
```

```
• _gameStart()
```

handles a START message from server by creating the board and passing it to GUI self the object pointer data the data TODO: wtf is data

# hasQuit()

Calls the board.RemovePlayer() method if a player quits the game.

#### **Parameters**

self the object pointer

dict data the data stream from client & server

\_jailed()

```
    ♣ _newTurn()
    def Client.Client._newTurn ( self, data )
    handles a TURN message from the server by telling GUI who's turn has begun
    Parameters
    self the object pointer data the data TODO: wtf is data
```

```
• _paid()
```

self the object pointer

```
def Client.Client._paid ( self, data )

update players balances as they change, 1 or more depending on transaction

Parameters

self the object pointer
dict data the data stream from client & server
```

```
◆ addToQueue()

def Client.Client.addToQueue ( self )

Adds data to the queue.

Parameters
```

## createGame()

```
def Client.Client.createGame ( self, username, password )
```

inform the server we wish to create a game

## • join()

```
def Client.Client.join ( self, address, username, password )
```

ask to join a specific game with username and password

#### **Parameters**

```
self the object pointer
```

int address the address (INT ARE YOU SURE?)

String username ...y'know, the user's name

String password ...uhh...

## listGames()

```
def Client.Client.listGames ( self )
```

Accessor: returns a list of games client has heard about.

### **Parameters**

self the object pointer

## poll()

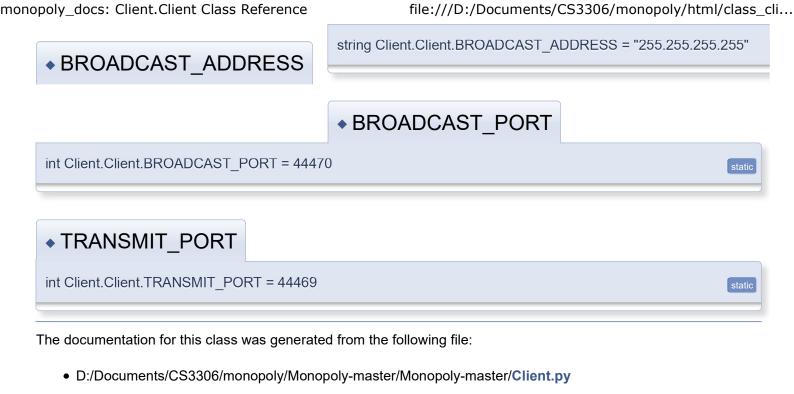
```
def Client.Client.poll ( self )
```

poll function

#### **Parameters**

self the object pointer





<u>doxy/gen</u> 1.8.13

Generated by