Table Of Content

Package controller

Class Summary

Controller

controller

Class Controller

```
java.lang.Object
    |
    +--controller.Controller
```

< Fields > < Constructors > < Methods >

public class **Controller**
extends java.lang.Object

Fields

network

`private NetworkHandler` **`network`**

running

`private boolean` **`running`**

session

`private Session` **`session`**

Constructors

Controller

`public` **`Controller`**`(Session session)`

Constructor with the session this controller talks to

Parameters:

session - - the session that the controller passes commands into

Methods

append

```
private byte[] append(byte[] data,
                     byte[] addition)
```

Parameters:

data - - first byte array
addition - - second byte array

Returns:

- a single byte array connected

---

append

```
private byte[] append(byte[] data,
                     java.lang.String addition)
```

Parameters:

data - - first byte array
addition - - string to be added

Returns:

- a single byte array connected

---

getDynamicData

```
private java.lang.String getDynamicData(byte[] data,
                                        int start)
```

Retrieves a given data from byte array where the start index is the size of the string to be received

Parameters:

data - - the byte[] array containing the information
start - - the index with the allocated size, start + 1 is where the string begins

Returns:

- the retrieved data

---

handshakeRequest

```
protected byte[] handshakeRequest(byte[] inFromClient)
```

CLIENT COMMAND - handshakeRequest

Parameters:

inFromClient - {'R' (1), IDlength (1), ID (x)}

Returns:

- the byte array to be returned

---

mediaPing

```
protected byte[] mediaPing(byte[] inFromClient)
```

CLIENT COMMAND - mediaPing

Parameters:

inFromClient - {'M' (1), 'P' (1)}

Returns:

- the byte array to be returned

---

mediaRequest

```
protected byte[] mediaRequest(byte[] inFromClient)
```

CLIENT COMMAND - mediaRequest

Parameters:

inFromClient - {'M' (1), 'R' (1), imageID (1), packet# (1)}

Returns:

- the byte array to be returned

---

parseNetworkCommand

```
public byte[] parseNetworkCommand(byte[] data)
              throws java.lang.IllegalArgumentException
```

CLIENT COMMAND CONTROL POINT - handles all incoming client commands

Parameters:

data - the byte array containing the command params

Returns:

- the byte array to be returned based on the initial command

Throws:

java.lang.IllegalArgumentException - - if the command given is invalid

start

```
public void start()
            throws java.net.SocketException
```

Starts the network socket to start accepting packets from clients

Throws:

java.net.SocketException - - if the port 9876 is in use

---

stop

```
public void stop()
            throws java.lang.IllegalArgumentException
```

Stops the network socket from accepting packets from clients

Throws:

java.lang.IllegalArgumentException - - if nothing is running

---

vote

```
protected byte[] vote(byte[] inFromClient)
```

CLIENT COMMAND - vote

Parameters:

inFromClient - {'V' (1), IDlength (1), ID (x), voteNumber (1), Votelength (1), Vote (x)}

Returns:

- the byte array to be returned

Package gui

Class Summary

ClientStage

The stage that displays the list of connected clients

ClientStage.ClientArea

TextFlow that will be used to display the updated client list

ConnectionInfo

The stage that will show connection info

ConsoleOutput

ConsoleStage

The output console for the project

HostScene

The active session scene for voting and changing what the clients see

SetupScene

SetupStage

The stage for creating the .voto files

VoteScene

VotoDesktop

VotoDesktopFX

The Main class in the project

VotoMenuBar

The Main MenuBar Wrapper for the Voto Desktop Application

gui

Class ClientStage

```
java.lang.Object
    |
    +--javafx.stage.Window
        |
        +--javafx.stage.Stage
            |
            +--gui.ClientStage
```

All Implemented Interfaces:
javafx.event.EventTarget

< Fields > < Constructors >

---

public class **ClientStage**
extends javafx.stage.Stage

The stage that displays the list of connected clients

Author:
Jay

Fields

activeScene

```
private javafx.scene.Scene activeScene
```

---

parent

```
private VotoMenuBar parent
```

---

session

```
private final Session session
```

Constructors

ClientStage

```
public  ClientStage(Session s,
                    VotoMenuBar parent)
```

Constructor that takes session pointer

Parameters:

s - Reference back to the running session

---

gui

Class ClientStage.ClientArea

```
java.lang.Object
    |
    +--javafx.scene.Node
        |
        +--javafx.scene.Parent
            |
            +--javafx.scene.layout.Region
                |
                +--javafx.scene.layout.Pane
                    |
                    +--javafx.scene.text.TextFlow
                        |
                        +--gui.ClientStage.ClientArea
```

All Implemented Interfaces:

java.awt.event.ActionListener, javafx.css.Styleable, javafx.event.EventTarget

---

< Fields > < Constructors > < Methods >

---

public class **ClientStage.ClientArea**
extends javafx.scene.text.TextFlow
implements java.awt.event.ActionListener

TextFlow that will be used to display the updated client list

Author:

Jay Uses a 1sec timer to constantly recall the list and redraw

Fields

t

```
private javax.swing.Timer t
```

Constructors

ClientArea

`public` **`ClientArea`**`()`

Constructor that makes 20/6 textarea and starts timer

Methods

actionPerformed

`public void` **`actionPerformed`**`(java.awt.event.ActionEvent e)`

When timer fires, recall client list and write out to the textarea

---

gui

Class ConnectionInfo

```
java.lang.Object
    |
    +--javafx.stage.Window
        |
        +--javafx.stage.Stage
            |
            +--gui.ConnectionInfo
```

All Implemented Interfaces:

javafx.event.EventTarget

< Constructors >

---

public class **ConnectionInfo**
extends javafx.stage.Stage

The stage that will show connection info

Author:

Josh

Constructors

ConnectionInfo

```
public  ConnectionInfo(java.lang.String n,
                       VotoMenuBar parent)
```

The constructor to create the new stage

Parameters:

n - The Session Name
parent - The menubar that created this

---

gui

Class ConsoleOutput

```
java.lang.Object
    |
    +--java.io.OutputStream
        |
        +--gui.ConsoleOutput
```

All Implemented Interfaces:
java.io.Closeable, java.io.Flushable

< Fields > < Constructors > < Methods >

---

public class **ConsoleOutput**
extends java.io.OutputStream

Fields

output

```
private final javafx.scene.control.TextArea output
```

---

sb

```
private final java.lang.StringBuilder sb
```

---

title

```
private java.lang.String title
```

Constructors

ConsoleOutput

```
public  ConsoleOutput(javafx.scene.control.TextArea out,
                      java.lang.String title)
```

Methods

close

```
public void close()
```

Overrides:

close in class java.io.OutputStream

---

flush

```
public void flush()
```

Overrides:

flush in class java.io.OutputStream

---

write

```
public void write(int b)
```

Writes out to the text area

Overrides:

write in class java.io.OutputStream

gui

Class ConsoleStage

```
java.lang.Object
    |
    +--javafx.stage.Window
        |
        +--javafx.stage.Stage
            |
            +--gui.ConsoleStage
```

All Implemented Interfaces:
javafx.event.EventTarget

< Fields > < Constructors >

---

public class **ConsoleStage**
extends javafx.stage.Stage

The output console for the project

Author:
Jay

Fields

activeScene

```
private javafx.scene.Scene activeScene
```

---

consoleOutput

```
private javafx.scene.control.TextArea consoleOutput
```

---

parent

```
private VotoMenuBar parent
```

Constructors

ConsoleStage

`public` **`ConsoleStage`**`(VotoMenuBar parent)`

Constructor that creates a 50/15 consolestage

Parameters:

parent - The menubar who enabled the console

---

gui

Class HostScene

```
java.lang.Object
    |
    +--javafx.scene.Scene
        |
        +--gui.HostScene
```

All Implemented Interfaces:

javafx.event.EventTarget

< Fields > < Constructors > < Methods >

---

public class **HostScene**
extends javafx.scene.Scene

The active session scene for voting and changing what the clients see

Author:

Nic / Josh

Fields

fc

`private javafx.stage.FileChooser` **`fc`**

---

file

`private java.io.File` **`file`**

---

mb

`private javafx.scene.control.MenuBar` **`mb`**

---

next

private javafx.scene.control.Button **next**

---

pane

private static javafx.scene.layout.Pane **pane**

---

picIndex

private int **picIndex**

---

picPane

private javafx.scene.control.ScrollPane **picPane**

---

pics

private javafx.scene.layout.HBox **pics**

---

questionIndex

private int **questionIndex**

---

questions

private java.util.ArrayList **questions**

---

rootHost

private static javafx.scene.layout.BorderPane **rootHost**

---

s

private Session **s**

Constructors

HostScene

```
public  HostScene(Session se,
                  double width,
                  double height,
                  VotoMenuBar mb)
```

The scene that will display the images and is essentially the active scene for the program

Parameters:

se - The active session for the scene
width - The width of the scene
height - The height of the scene
mb - The menubar to be passed in with the scene

Methods

addImgToSP

```
private void addImgToSP(javafx.scene.image.ImageView iViewPrev)
```

Adds image to the HBox

Parameters:

iViewPrev - The image to be added as ImageView

---

addPic

```
private void addPic(java.lang.String filepath)
```

Opens picture and loads into an ImageView

Parameters:

filepath - The filepath of the image to be loaded

---

nextQuestion

```
private void nextQuestion()
```

Moves onto the nextQuestion in the session

---

openFile

```
private void openFile()
```

Open picture from file chooser to host pane

gui

Class SetupScene

```
java.lang.Object
    |
    +--javafx.scene.Scene
        |
        +--gui.SetupScene
```

All Implemented Interfaces:
javafx.event.EventTarget

< Fields > < Constructors > < Methods >

---

public class **SetupScene**
extends javafx.scene.Scene

Fields

ansButton

`javafx.scene.control.Button[] ansButton`

---

exitItem

`private javafx.scene.control.MenuItem exitItem`

---

fc

`private javafx.stage.FileChooser fc`

---

file

`private java.io.File file`

---

fileMenu

`private javafx.scene.control.Menu fileMenu`

---

fileName

private java.lang.String **fileName**

---

menuBar

private javafx.scene.control.MenuBar **menuBar**

---

openItem

private javafx.scene.control.MenuItem **openItem**

---

output

private java.io.BufferedWriter **output**

---

picIndex

private int **picIndex**

---

picPane

private javafx.scene.control.ScrollPane **picPane**

---

pics

private javafx.scene.layout.HBox **pics**

---

rootSetup

private static javafx.scene.layout.BorderPane **rootSetup**

---

saveItem

private javafx.scene.control.MenuItem **saveItem**

Constructors

SetupScene

```
public  SetupScene(double width,
                   double height)
```

Constructor to create the setup scene

Parameters:

width - Scene width
height - Scene height

Methods

addImgToSP

```
private void addImgToSP(javafx.scene.image.ImageView iViewPrev)
```

---

addMenu

```
private void addMenu()
```

---

addPic

```
private void addPic(java.lang.String filepath)
```

---

answerPane

```
private void answerPane()
```

---

openFile

```
private void openFile()
```

Loads a selected files in filechooser

---

saveFile

```
private void saveFile()
```

---

gui

Class SetupStage

```
java.lang.Object
    |
    +--javafx.stage.Window
        |
        +--javafx.stage.Stage
            |
            +--gui.SetupStage
```

All Implemented Interfaces:
javafx.event.EventTarget

< Fields > < Constructors > < Methods >

---

public class **SetupStage**
extends javafx.stage.Stage

The stage for creating the .voto files

Author:
Nic

Fields

ansButton

```
javafx.scene.control.Button[] ansButton
```

---

exitItem

```
private javafx.scene.control.MenuItem exitItem
```

---

fc

```
private javafx.stage.FileChooser fc
```

---

file

```
private java.io.File file
```

---

fileMenu

```
private javafx.scene.control.Menu fileMenu
```

---

fileName

```
private java.lang.String fileName
```

---

menuBar

```
private javafx.scene.control.MenuBar menuBar
```

---

openItem

```
private javafx.scene.control.MenuItem openItem
```

---

outFile

```
private java.io.File outFile
```

---

output

```
private java.io.BufferedWriter output
```

---

picIndex

```
private int picIndex
```

---

picPane

```
private javafx.scene.control.ScrollPane picPane
```

---

pics

```
private javafx.scene.layout.HBox pics
```

---

rootSetup

```
private javafx.scene.layout.BorderPane rootSetup
```

---

saveItem

```
private javafx.scene.control.MenuItem saveItem
```

Constructors

SetupStage

```
public  SetupStage()
```

Constructor for the setup stage

Methods

addImgToSP

```
private void addImgToSP(javafx.scene.image.ImageView iViewPrev)
```

Add imageview to the Hbox

Parameters:

iViewPrev - The imageview to be added

---

addMenu

```
private void addMenu()
```

Basic menubar for this stage

---

addPic

```
private void addPic(java.lang.String filepath)
```

Converts image to ImageView and loads into HBox

Parameters:

filepath - The image filepath

---

answerPane

```
private void answerPane()
```

The Answer Pane that allows you to select the correct answer

---

openFile

```
private void openFile()
```

Loads specified file through a filechooser

---

saveFile

```
private void saveFile()
```

Saves the current session voto file

---

gui

Class VoteScene

```
java.lang.Object
    |
    +--javafx.scene.Scene
        |
        +--gui.VoteScene
```

All Implemented Interfaces:
javafx.event.EventTarget

< Fields > < Constructors >

---

public class **VoteScene**
extends javafx.scene.Scene

Fields

joinGrid

```
private javafx.scene.layout.GridPane joinGrid
```

---

rootJoin

```
private static javafx.scene.layout.BorderPane rootJoin
```

---

votingButtons

```
private javafx.scene.control.Button[] votingButtons
```

Constructors

VoteScene

```
public  VoteScene(double width,
                  double height)
```

---

gui

Class VotoDesktop

```
java.lang.Object
    |
    +--gui.VotoDesktop
```

All Implemented Interfaces:
java.awt.event.ActionListener, java.lang.Runnable

< Fields > < Constructors > < Methods >

---

public class **VotoDesktop**
extends java.lang.Object
implements java.awt.event.ActionListener, java.lang.Runnable

Fields

c

```
private Controller c
```

---

connectButton

```
private javax.swing.JButton connectButton
```

---

f

```
private javax.swing.JFrame f
```

---

fileChooser

```
private javax.swing.JFileChooser fileChooser
```

---

hostButton

```
private javax.swing.JButton hostButton
```

---

hostPanel

```
private javax.swing.JPanel hostPanel
```

---

ipField

```
private javax.swing.JTextField ipField
```

---

ipLabel

```
private javax.swing.JLabel ipLabel
```

---

joinButton

```
private javax.swing.JButton joinButton
```

---

openButton

```
private javax.swing.JButton openButton
```

---

s

```
private Session s
```

---

startPanel

```
private javax.swing.JPanel startPanel
```

---

t

```
private javax.swing.Timer t
```

Constructors

VotoDesktop

```
public  VotoDesktop()
```

Methods

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent e)
```

---

hostGUI

```
private void hostGUI()
```

---

joinGUI

```
private void joinGUI()
```

---

main

```
public static void main(java.lang.String[] args)
```

---

run

```
public void run()
```

---

startGUI

```
private void startGUI()
```

---

gui

Class VotoDesktopFX

```
java.lang.Object
    |
    +--javafx.application.Application
        |
        +--gui.VotoDesktopFX
```

< Fields > < Constructors > < Methods >

---

public class **VotoDesktopFX**
extends javafx.application.Application

The Main class in the project

Author:
Nic

Fields

IP

```
protected static java.lang.String IP
```

---

hostButton

```
private javafx.scene.control.Button hostButton
```

---

joinButton

```
private javafx.scene.control.Button joinButton
```

---

menu

```
private VotoMenuBar menu
```

---

primary

```
private javafx.stage.Stage primary
```

---

root

```
private javafx.scene.layout.BorderPane root
```

---

s

```
private Session s
```

---

setupButton

```
private javafx.scene.control.Button setupButton
```

Constructors

VotoDesktopFX

```
public  VotoDesktopFX()
```

Methods

exitProgram

```
protected void exitProgram()
```

Quit

---

hostGUI

```
protected void hostGUI()
```

Host GUI displays IP address, allows user to open pictures, displays pictures, and lets the user select the correct answer for each picture

---

joinGUI

```
private void joinGUI(javafx.stage.Stage p)
```

---

main

```
public static void main(java.lang.String[] args)
```

---

start

```
public void start(javafx.stage.Stage primaryStage)
```

Start GUI has host or join options

Overrides:

start in class javafx.application.Application

---

gui

Class VotoMenuBar

```
java.lang.Object
    |
    +--javafx.scene.Node
        |
        +--javafx.scene.Parent
            |
            +--javafx.scene.layout.Region
                |
                +--javafx.scene.control.Control
                    |
                    +--javafx.scene.control.MenuBar
                        |
                        +--gui.VotoMenuBar
```

All Implemented Interfaces:

javafx.css.Styleable, javafx.event.EventTarget, javafx.scene.control.Skinnable

---

< Fields > < Constructors > < Methods >

---

public class **VotoMenuBar**
extends javafx.scene.control.MenuBar

The Main MenuBar Wrapper for the Voto Desktop Application

Author:

Josh

Fields

clientsItem

```
private javafx.scene.control.MenuItem clientsItem
```

---

connectionItem

```
private javafx.scene.control.MenuItem connectionItem
```

---

consoleItem

```
private javafx.scene.control.MenuItem consoleItem
```

---

exitItem

```
private javafx.scene.control.MenuItem exitItem
```

---

fileMenu

```
private javafx.scene.control.Menu fileMenu
```

---

graphItem

```
private javafx.scene.control.MenuItem graphItem
```

---

newItem

```
private javafx.scene.control.MenuItem newItem
```

---

nextItem

```
private javafx.scene.control.MenuItem nextItem
```

---

openItem

```
private javafx.scene.control.MenuItem openItem
```

parent

```
private VotoDesktopFX parent
```

saveItem

```
private javafx.scene.control.MenuItem saveItem
```

sessionMenu

```
private javafx.scene.control.Menu sessionMenu
```

windowMenu

```
private javafx.scene.control.Menu windowMenu
```

Constructors

VotoMenuBar

```
public  VotoMenuBar(Session s,
                    VotoDesktopFX p)
```

The universal menu bar for the program

Parameters:

s - The active session
p - The main stage

Methods

enableClient

```
public void enableClient()
```

Re-enable the client dropdown item

enableConnection

```
public void enableConnection()
```

Re-enable the connection dropdown item

enableConsole

`public void` **`enableConsole`**`()`

Re-enable the console dropdown item

---

setNext

`public void` **`setNext`**`(javafx.event.EventHandler value)`

Adjust what the next dropdown item does

Parameters:

value - The ActionEven to happen

---

setOpenFile

`public void` **`setOpenFile`**`(javafx.event.EventHandler value)`

Change open to do something else

Parameters:

value - The new ActionEvent

Package networking

Class Summary

NetworkHandler

UDPSocket

---

networking

Class NetworkHandler

```
java.lang.Object
    |
    +--networking.NetworkHandler
```

All Implemented Interfaces:
java.io.Closeable, java.lang.Runnable

< Fields > < Constructors > < Methods >

---

public class **NetworkHandler**
extends java.lang.Object
implements java.io.Closeable, java.lang.Runnable

Author:

zomby This class controls whats coming in and out of the UDPSocket onPacketReceived, it passes it to the parser and finds the proper command

Fields

control

`private Controller` **`control`**

---

socket

`private UDPSocket` **`socket`**

Constructors

NetworkHandler

```
public  NetworkHandler(Controller control)
        throws java.net.SocketException
```

Create the controller

Throws:

java.net.SocketException - - If something is already using port 9876

Methods

close

```
public void close()
```

Close the socket

---

onPacketReceived

```
public void onPacketReceived(java.net.DatagramPacket inFromClient)
```

Parses the DatagramPacket into a set of keyword arguments, passes them onto a command parser

Parameters:

inFromClient - - The datagram packet received from client

---

reply

```
public void reply(byte[] data,
                  java.net.DatagramPacket in)
```

Replies the given byte array to the location of the datagram packet

Parameters:

data - - the byte array to be sent
in - - the datagram packet to have the byte array sent too

---

run

```
public void run()
```

Start the socket

networking

Class UDPSocket

```
java.lang.Object
    |
    +--networking.UDPSocket
```

All Implemented Interfaces:
java.io.Closeable, java.lang.Runnable

< Fields > < Constructors > < Methods >

---

public class **UDPSocket**
extends java.lang.Object
implements java.io.Closeable, java.lang.Runnable

Author:

zomby This class is designed to open a Datagram socket on a defined port and then both send and receive

Fields

PORT

```
private final int PORT
```

---

isListening

```
private volatile boolean isListening
```

---

listener

```
private NetworkHandler listener
```

---

socket

```
private java.net.DatagramSocket socket
```

Constructors

UDPSocket

`public` **`UDPSocket`**`(NetworkHandler l)`

Create a new datagram socket, catch the error of something else using the port.

Methods

close

`public void` **`close`**`()`

Wait a second for the isListening to take affect then close it cause the loop in run will have stopped This is not necessary but guarantees no error

---

isListening

`public boolean` **`isListening`**`()`

Is the socket still listening

Returns:

If the socket is still listening or not

---

run

`public void` **`run`**`()`

Start listening and in an infinite loop constantly receive packets, sending them up to the listener.

---

send

`public void` **`send`**`(java.net.DatagramPacket outToClient)`

Send DatagramPacket to client

Parameters:

outToClient - - Datagram packet to be sent

Package session

Class Summary

Client

The client class which holds votes and IDs

Question

The Question class which holds the active image data, vote list, choices, and answers

QuestionData

The Question class stripped into just the important data to remove usage of RAM

Session

The session class which holds the question list, and is the "Active" class

Vote

A vote object

---

session

Class Client

```
java.lang.Object
    |
    +--session.Client
```

< Fields > < Constructors > < Methods >

---

public class **Client**
extends java.lang.Object

The client class which holds votes and IDs

Author:

Jay

Fields

ID

```
private java.lang.String ID
```

---

lastVote

```
private Vote lastVote
```

voteList

```
private java.util.ArrayList voteList
```

voteNum

```
public int voteNum
```

Constructors

Client

```
public  Client(java.lang.String clientID)
```

Client constructor

Parameters:

clientID - - this client's ID

Methods

equals

```
public boolean equals(java.lang.Object o)
```

Tells whether or not the sent arg is the current client

Parameters:

ID - - a client ID to validate

Returns:

True if the ID sent in matches the current client's ID False if the IDs don't match

Overrides:

equals in class java.lang.Object

getClientVoteList

```
public java.lang.Iterable getClientVoteList()
```

Returns a list of all the clients final votes for current session

Returns:

List of client votes

getID

`public java.lang.String **getID**()`

Gets the client ID

Returns:

the client's ID as a string

---

getLastVote

`public Vote **getLastVote**()`

Returns the last vote sent by the specified client

Returns:

last received vote

---

setAnswerVote

`public void **setAnswerVote**(Question q, Vote v)`

Sets the final vote submitted for the question

Parameters:

v - The final vote submitted

---

setLastVote

`public void **setLastVote**(Vote newVote)`

Sets the clients most recently sent vote as their current vote

Parameters:

lastVote - - the most recently sent vote
oldVote - - reference to previous client vote to be discarded

---

session

Class Question

```
java.lang.Object
    |
    +--session.Question
```

< Fields > < Constructors > < Methods >

---

public class **Question**
extends java.lang.Object

The Question class which holds the active image data, vote list, choices, and answers

Author:
Nick

Fields

answer

```
private Vote answer
```

---

answerSet

```
private java.util.HashMap answerSet
```

---

choices

```
private java.util.HashMap choices
```

---

imageID

```
private int imageID
```

---

questionImg

```
public java.util.ArrayList questionImg
```

Constructors

Question

```
public  Question(java.util.ArrayList img,
                 int imageID)
```

Question constructor

Parameters:

s - - current session of this question
img - - image loaded with this question
imageID - - image ID for image param

Methods

addVote

```
public boolean addVote(Client client,
                       java.lang.String clientVote,
                       int voteNum)
```

Adds a vote from a client to the current question

Parameters:

clientID - - ID of the client sending the vote
clientVote - - the actual vote sent by the client

---

endQuestion

```
public void endQuestion()
```

Ends the current question and sets each client's last vote to be their final recorded vote

---

getAnswer

```
public Vote getAnswer()
```

Returns the correct answer for this question

Returns:

the Vote corresponding to the correct answer

---

getAnswerSet

```
public java.util.HashMap getAnswerSet()
```

Returns the mapping of clients to their votes for the current question

Returns:

HashMap of votes listing the clients who selected it

---

getChoices

```
public java.util.HashMap getChoices()
```

---

getImagePacket

```
public byte[] getImagePacket(int packetNum)
              throws java.lang.IllegalArgumentException
```

Returns a byte array containing the image for the current question

Parameters:

packetNum - - the packet number for the desired image

Returns:

image byte array

Throws:

java.lang.IllegalArgumentException -

---

imageID

```
public int imageID()
```

Returns the image ID for current question

Returns:

the image ID

---

imageSize

```
public int imageSize()
```

Returns the size of the current questions image

Returns:

current image size

setAnswer

```
public void setAnswer(java.lang.String ans)
```

Sets the correct answer for current question

Parameters:

ans - - String for the correct answer

---

session

Class QuestionData

```
java.lang.Object
    |
    +--session.QuestionData
```

< Fields > < Constructors > < Methods >

---

public class **QuestionData**
extends java.lang.Object

The Question class stripped into just the important data to remove usage of RAM

Author:

Nick

Fields

ID

```
private int ID
```

---

questionAnswerData

```
public java.util.HashMap questionAnswerData
```

Constructors

QuestionData

```
public  QuestionData(Question q,
                     int qID)
```

QuestionData constructor

Methods

equals

```
public boolean equals(java.lang.Object o)
```

Overrides:

equals in class java.lang.Object

---

getID

```
public int getID()
```

Checks for a given ID and if there is available question data for it

Returns:

- the ID we are trying to access question data for

---

session

Class Session

```
java.lang.Object
    |
    +--session.Session
```

< Fields > < Constructors > < Methods >

---

public class **Session**
extends java.lang.Object

The session class which holds the question list, and is the "Active" class

Author:

Jay

Fields

ID

```
public java.lang.String ID
```

---

clientList

```
private java.util.ArrayList clientList
```

---

control

```
private Controller control
```

---

currentQuestion

```
private Question currentQuestion
```

---

dataList

```
private java.util.ArrayList dataList
```

---

imageID

```
private int imageID
```

---

questionList

```
private java.util.ArrayList questionList
```

Constructors

Session

```
public  Session(java.lang.String ID)
```

Methods

addClient

```
public boolean addClient(java.lang.String ID)
```

Adds a new client to the session's client list with their ID

Parameters:

ID - - the new client's ID

---

addClientVote

```
public boolean addClientVote(java.lang.String clientID,
                             java.lang.String vote,
                             int voteNum)
```

Adds a client vote to the current Question

Parameters:

clientID - The string of the clients ID
vote - The vote string

Returns:

false if not added or invalid, true if added

---

getClient

```
public Client getClient(java.lang.String clientID)
```

Returns a client object based off of the client ID passed in

Parameters:

clientID - - ID of the desired client

Returns:

a client object

---

getClients

```
public java.lang.Iterable getClients()
```

---

getCurrentImageID

```
public int getCurrentImageID()
```

Returns the ID for the image of the current question

Returns:

ID of current question image

---

getCurrentImagePacketCount

`public int` **`getCurrentImagePacketCount`**`()`

Returns the number of packets for the current question image

Returns:

image packet count

---

getCurrentImageSize

`public int` **`getCurrentImageSize`**`()`

Returns the size (in bytes) of the current session image

Returns:

image size in bytes

---

getCurrentQuestion

`public` `Question` **`getCurrentQuestion`**`()`

the current Question

Returns:

the current quesion

---

getImagePacket

```
public byte[] getImagePacket(int imageID,
                             int packetNumber)
              throws java.lang.IllegalArgumentException
```

Returns the image packet for the current question of the session

Parameters:

imageID - - ID of the question image
packetNumber - - corresponding packet number

Returns:

byte array packet for the current image

Throws:

java.lang.IllegalArgumentException -

---

hasImage

```
public boolean hasImage()
```

If a current Question is loaded

Returns:

- True if loaded, false if not (null)

---

loadImage

```
public java.util.ArrayList loadImage(java.lang.String filename)
                          throws java.io.IOException
```

Loads an image into an arraylist of bytearray of 64KB each

Parameters:

filename - - The filename of the image to be loaded

Returns:

- An arraylist of 60KB byte arrays

Throws:

java.io.IOException - - if the filename is invalid

---

returnQuestionDataBarChart

```
public java.util.HashMap returnQuestionDataBarChart(int ID)
```

Returns a HashMap used to build a bar chart corresponding to votes per choice for a given question

Parameters:

ID - - ID for the question we are accessing data for

Returns:

- Each vote choice with its corresponding number of votes

---

returnQuestionDataPieChart

```
public java.util.HashMap returnQuestionDataPieChart(int ID)
```

Returns a HashMap used to build a pie chart corresponding to the percentage of votes per choice of a given question

Parameters:

ID - - ID for the question we are accessing data for

Returns:

- Each vote choice with its corresponding percentage of votes

setCurrentQuestion

```
public boolean setCurrentQuestion(java.lang.String filename,
                                  java.lang.String answer)
```

Sets a new Question for the Session. Loads image via filename and sets the correct answer to be that of the inputted string.

Parameters:

filename - The image file to be loaded
answer - The answer string to be accepted

Returns:

True if loads properly, False if image fails

---

setID

```
public void setID(java.lang.String id)
```

---

start

```
public void start()
            throws java.net.SocketException
```

Starts a Voto session

Throws:

java.net.SocketException -

---

stop

```
public void stop()
            throws java.lang.IllegalArgumentException
```

Stops (or ends) the current Voto session

Throws:

java.lang.IllegalArgumentException -

session

Class Vote

```
java.lang.Object
    |
    +--session.Vote
```

< Fields > < Constructors > < Methods >

---

public class **Vote**
extends java.lang.Object

A vote object

Author:

Jay

Fields

ID

```
private java.lang.String ID
```

Constructors

Vote

```
public  Vote(java.lang.String i)
```

Vote constructor

Parameters:

i - - ID for the vote object

Methods

equals

```
public boolean equals(java.lang.Object o)
```

Overrides:

equals in class java.lang.Object

---

getID

```
public java.lang.String getID()
```

INDEX

I

J

L

M

N

O

P

Q

R

S

T

U

V

W