**<pre>**

**JokeServer.java**

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.PrintStream;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**import** java.util.ArrayList;

**import** java.util.Random;

/\*\*--------------------------------------------------------

1. **@author** Prashant Padmashali **@Date** 27, September 2015

2. Java version used, if not the official version for the class:

**@Filename** JokeServer.java

**@version** 1.8.0\_60

**@IDE\_Version** Version: Neon Milestone 1 (4.6.0M1) Build id: 20150820-1211

3. Precise command-line compilation examples / instructions:

**@set** path="<drive>:\<path>\Java\jdk\bin"

**@set** classpath=%classpath%;.;

javac JokeServer.java

java JokeServer

4. Precise examples / instructions to run this program:

> javac JokeServer.java

> java JokeServer

> javac JokeClient.java

> java JokeClient

> java JokeClientAdmin.java

> java JokeClientAdmin

All acceptable commands are displayed on the various consoles.

This runs across machines, in which case you have to pass the IP address of

the server to the clients. For exmaple, if the server is running at

140.192.1.22 then you would type:

> java JokeClient 140.192.1.22

> java JokeClientAdmin 140.192.1.22

5. List of files needed for running the program.

a. checklist.html

b. JokeServer.java

c. JokeClient.java

d. JokeClientAdmin.java

5. Notes:

The Program is a joke server which is built with the help of InetServer.java and also with the code-snippet

provided my Professor. Elliott's in "http://condor.depaul.edu/elliott/435/hw/programs/joke/joke-threads.html"

also the socket implementation of inetserver and inetclient is used as reference(http://condor.depaul.edu/elliott/435/hw/programs/inet/InetPDF-C.pdf)

to setup the client and server

The program sends jokes and proverbs at random order without repetition to the client. The code is not optimized.

----------------------------------------------------------\*\*/

**public** **class** JokeServer {

/\*\*

\*controlSwitch variable makes sure that all the input from client is being continuously received.

\*port variable is the port number in which the server will be running and client will be listening.

\*sMode="joke" is to keep it as a default mode

\*/

**public** **static** **boolean** *clcontolSwitch*=**true**;

**public** **static** String *sMode*="joke";

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

// **TODO** Auto-generated method stub

**int** q\_len=6;/\* Number of requests for OpSys to queue \*/

**int** port =4141;

Socket sock; //socket connection setup

AdminLooper AL= **new** AdminLooper(); // creating a different thread for server to listen to AdminLooper

Thread t=**new** Thread(AL);

t.start(); // ...and start it, waiting for administration input

//initiating a socket to connect to client

ServerSocket serversock =**new** ServerSocket(port, q\_len);

System.***out***.println("Prashant's Joke Server starting up, listening for client port at "+port+"\n");

**while**(*clcontolSwitch*){

sock=serversock.accept(); //to accept a new connection from client and following new upcoming connections

**if**(*clcontolSwitch*) **new** JokeworkerThread(sock).start(); //worker is assigned to the new accepted connection

**try**{Thread.*sleep*(1000);} **catch**(InterruptedException ex){}

}serversock.close();

}

}

**class** JokeworkerThread **extends** Thread {

Socket sock;

/\*\*

\* **@jokes** - ArrayList initialization for jokes

\* **@proverb** - ArrayList initialization for proverbs

\* **@jokesCopy** and **@ProverbsCopy.**- ArrayList used to check for repetition of the jokes and proverbs.

\*/

**private** **static** ArrayList<String> *jokes* = **new** ArrayList<String>();

**private** **static** ArrayList<String> *jokesCopy* = **new** ArrayList<String>();

**private** **static** ArrayList<String> *proverbs* = **new** ArrayList<String>();

**private** **static** ArrayList<String> *proverbsCopy* = **new** ArrayList<String>();

**private** **static** String *username*;

JokeworkerThread (Socket s) {sock =s;}

PrintStream out = **null**;

BufferedReader in = **null**;

**static** **boolean** *flag* =**true**;

**public** **void** run(){

**try**{

in=**new** BufferedReader (**new** InputStreamReader(sock.getInputStream()));

out=**new** PrintStream(sock.getOutputStream());

**try** {

String name=in.readLine();

*jokeproverbContainer*(name, out);

}**catch** (IOException x){

System.***out***.println("Mayday! Mayday! Server read error");

x.printStackTrace();

}

sock.close();

}**catch**(IOException ioe) {System.***out***.println(ioe);}

}

/\*\***@username** - the name entered by the user

\***@selectedJoke** - the randomly selected joke

\***@selectedProverb** - the randomly selected proverb

Based on user selection displays either a proverb or a joke\*\*/

**static** **void** jokeproverbContainer(String name, PrintStream out) {

*username*="";

*username*= name;

//@sMode - checks for Server Mode to be joke, and accordingly sends a joke

**if** (JokeServer.*sMode*.equalsIgnoreCase("joke")){

//out.println(jokes[Integer.parseInt(pick)] + "\n");

String selectedJoke = *selectrandomlyFromJokes*();

out.println(selectedJoke);

}

//@sMode - checks for Server Mode to be proverb, and accordingly sends a proverb

**else** **if** (JokeServer.*sMode*.equalsIgnoreCase("proverb")) {

String selectedProverb = *selectrandomlyFromProverbs*();

out.println(selectedProverb);

//out.println(proverbs[Integer.parseInt(pick)] + "\n");

}

**if**(JokeServer.*sMode*.equalsIgnoreCase("Maintenance")){

out.print("Server is down due to Maintenance");

}

}

//@jokes - list of all jokes

//@jokesCopy - jokes which have been displayed already

// Removes the already displayed jokes in order to avoid repetition

**static** **void** removeDuplicatesFromJokes(){

**for** (String joke : *jokesCopy*) {

**if** (*jokes*.contains(joke)) {

*jokes*.remove(joke);

}

}

}

//@jokes - list of all jokes

//@jokesCopy - jokes which have been displayed already

// Removes the already displayed proverbs in order to avoid repetition

**static** **void** removeDuplicatesFromProverbs(){

**for** (String proverb : *proverbsCopy*) {

**if** (*proverbs*.contains(proverb)) {

*proverbs*.remove(proverb);

}

}

}

//@jokes - list of all jokes

//@jokesCopy - jokes which have been displayed already

//@username - the name entered by the user

//@random - a new instance of random

//@pick - selects a random number from 0 to jokes.size()

//@joke - randomly chosen joke

// This method chooses a random joke from the list of available jokes

// The jokes are not repeated until the list gets exhausted

// JokesCopy is cleared when jokeslist completes the set of jokes

**static** String selectrandomlyFromJokes(){

*jokes*.clear();

*jokes*.add("A.J>"+*username*+", Error, no keyboard. Press F1 to continue.");

*jokes*.add("B.J>"+*username*+", Ever notice how fast Windows runs? Neither did I.");

*jokes*.add("C.J>"+*username*+", Everyone has a photographic memory. Some don't have film." );

*jokes*.add("D.J>"+*username*+", If you lend someone $20, and never see that person again; it was probably worth it.");

*jokes*.add("E.J>"+*username*+", Don't drink and drive. You might hit a bump and spill your drink.");

*removeDuplicatesFromJokes*();

//Randomizing of jokes by using random package of java

Random random=**new** Random();

**int** pick = random.nextInt(*jokes*.size());

String joke = *jokes*.get(pick);

*jokesCopy*.add(joke);

// Reset jokescopy when all jokes have been selected atleast once

**int** jokesCopySize = *jokesCopy*.size();

**if** (jokesCopySize==5) {

*jokesCopy*.clear();

}

**return** joke;

}

/\*\*

\* **@proverbs** - list of all proverbs

\* **@proverbsCopy** - proverbs which have been displayed already

\* **@username** - the name entered by the user

\* **@random** - a new instance of random

\* **@pick** - selects a random number from 0 to proverbs.size()

\* **@proverb** - randomly chosen joke

\* This method chooses a random proverb from the list of available proverbs

\* The proverbs are not repeated until the list gets exhausted

\* JproverbsCopy is cleared when proverbs list completes the set of jokes \*\*/

**static** String selectrandomlyFromProverbs(){

*proverbs*.clear();

*proverbs*.add("A.P>If you want something done right, "+*username*+" you have to do it yourself.");

*proverbs*.add("B.P>"+*username*+" One man's trash is another man's treasure.");

*proverbs*.add("C.P>"+*username*+" Don't count your chickens before they hatch.");

*proverbs*.add("D.P>"+*username*+" There's no such thing as a free lunch.");

*proverbs*.add("E.P>"+*username*+" When the going gets tough, the tough get going.");

*removeDuplicatesFromProverbs*();

//Randomizing of proverbs by using random package of java

Random random=**new** Random();

**int** pick = random.nextInt(*proverbs*.size());

String proverb = *proverbs*.get(pick);

*proverbsCopy*.add(proverb);

// Reset proverbscopy when all jokes have been selected atleast once

**int** proverbsCopySize = *proverbsCopy*.size();

**if** (proverbsCopySize==5) {

*proverbsCopy*.clear();

}

**return** proverb;

}

}

/\*\*

\*

\* **@author** ppadm

\* It helps JokeAdminClient.java to connect

\* **@port** - to connect to adminclient on a different port

\*/

**class** AdminLooper **implements** Runnable{

**public** **static** **boolean** *adminControlSwitch*=**true**;

**public** **void** run(){ // Running the Admin listen loop

System.***out***.println("In the admin looper thread");

**int** q\_len=6;

**int** port=4242;

Socket sock;

**try**{

ServerSocket servsock= **new** ServerSocket(port,q\_len);

**while**(*adminControlSwitch*){

// wait for the next ADMIN client connection:

sock=servsock.accept();

**new** AdminWorker(sock).start();

}servsock.close();

}**catch**(IOException ioe) {System.***out***.println(ioe);}

}

}

//Admin worker thread to handle new connection from adminClient

**class** AdminWorker **extends** Thread{

Socket sock;

//private Object sd;

AdminWorker(Socket s) {sock =s;}

**public** **void** run(){

PrintStream out =**null**;

BufferedReader in =**null**;

//initiating input stream to read and output stream to print on the socket connection

**try**{

in =**new** BufferedReader(**new** InputStreamReader(sock.getInputStream()));

out=**new** PrintStream(sock.getOutputStream());

**if**(JokeServer.*clcontolSwitch* != **true** ) {

System.***out***.println("Listener is now shutting down as per the client request");

out.println("Server is now shutting down. BYE! BYE! ");

}

**else** **try** {

String smode;

//String shutdown;

//shutdown=sd;

smode=in.readLine();

**if** (smode.equalsIgnoreCase("shutdown")) { //to shutdown the server

JokeServer.*clcontolSwitch* = **false**;

System.***out***.println("Admin Worker has captured a shutdown request");

out.println("Please send the final shutdown request to the Server");

out.println("shutdown request has been noted by the worker");

System.*exit*(0);

}

**if**(smode.equalsIgnoreCase("joke")) {//to change the server to joke mode

JokeServer.*sMode*="joke";

System.***out***.println("Mode change to JOKES has been captured");

out.println("Mode change has been completed to Joke");

out.println("Mode = JOKES");

}

**if**(smode.equalsIgnoreCase("proverb")) {//to change the server to proverb mode

JokeServer.*sMode*="proverb";

System.***out***.println("Mode change to PROVERBS has been captured");

out.println("Mode change has been completed to Proverb");

out.println("Mode = PROVERBS");

}

**if**(smode.equalsIgnoreCase("maintenance")) {//to change the server to maintenance mode

JokeServer.*sMode*="maintenance";

System.***out***.println("Mode change to MAINTENANCE has been captured");

out.println("Mode change has been completed to Maintenance");

out.println("Mode = MAINTENANCE");

}

**else**{

out.println("Error code:400 Bad Request");

}

}**catch**(IOException x){

System.***out***.println("Server read error");

x.printStackTrace();

}

sock.close();

}**catch** (IOException ioe){System.***out***.println(ioe);}

}

}

**JokeClient**

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.PrintStream;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**import** java.util.ArrayList;

**import** java.util.Random;

/\*\*--------------------------------------------------------

1. **@author** Prashant Padmashali **@Date** 27, September 2015

2. Java version used, if not the official version for the class:

**@Filename** JokeClient.java

**@version** 1.8.0\_60

**@IDE\_Version** Version: Neon Milestone 1 (4.6.0M1) Build id: 20150820-1211

3. Precise command-line compilation examples / instructions:

**@set** path="<drive>:\<path>\Java\jdk\bin"

**@set** classpath=%classpath%;.;

javac JokeClient.java

java JokeClient

4. Precise examples / instructions to run this program:

> javac JokeServer.java

> java JokeServer

> javac JokeClient.java

> java JokeClient

> java JokeClientAdmin.java

> java JokeClientAdmin

All acceptable commands are displayed on the various consoles.

This runs across machines, in which case you have to pass the IP address of

the server to the clients. For exmaple, if the server is running at

140.192.1.22 then you would type:

> java JokeClient 140.192.1.22

> java JokeClientAdmin 140.192.1.22

5. List of files needed for running the program.

a. checklist.html

b. JokeServer.java

c. JokeClient.java

d. JokeClientAdmin.java

5. Notes:

The Program is a joke Client which is built with the help of InetClient.java also the socket implementation of inetserver and inetclient is used as reference

(http://condor.depaul.edu/elliott/435/hw/programs/inet/InetPDF-C.pdf)the Jokeclient is the end user application to which the

joke or proverb is sent according to the set mode of the server, By default the server is in joke mode

----------------------------------------------------------\*\*/

@SuppressWarnings("unused")

**public** **class** JokeClient {

//code moved to server during code revision

//private static ArrayList<String> jokes;

//private static ArrayList<String> jokesCopy;

/\*\*

\* **@param** args

\*/

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

// **TODO** Auto-generated method stub

String serverName; //@serverName- localhost name of the system

//InetAddress machine=InetAddress.getLocalHost();

**if** (args.length < 1) serverName = "localhost";

**else** serverName = args[0];

System.***out***.println("Prashant's Joke client \n");

System.***out***.println("Using Server "+serverName+", port 4141");

BufferedReader in=**new** BufferedReader(**new** InputStreamReader(System.***in***));

//code moved to server during code revision

//jokes = new ArrayList<String>(5);

//jokesCopy = new ArrayList<String>(5);

System.***out***.println("Please enter your name. ");

System.***out***.flush();

String name; // storing user entered name

name=in.readLine();

**try**{

String looper;

// loop for next joke or proverb or to quit. depending on user input

**do**{

System.***out***.println("enter 'quit' to close the application or Hit 'enter' key to proceed");

System.***out***.flush();

looper=in.readLine();

**if**(!(looper.equals("quit")))

*getRemoteAddress*(name,serverName);

}**while** (!(looper.equals("quit")));

System.***out***.println("Cancelled by user request.");

}**catch**(IOException x) {x.printStackTrace();}

}

/\*\*

\* **@sock**- variable is for establishing a new socket connection.

\* **@toServer**- variable is to send the user input domain to server.

\* **@fromServer**- variable is to get back the output from the server as IP of the domain.

\*\*/

**static** **void** getRemoteAddress( String name, String serverName) {

Socket sock;

BufferedReader fromServer;

PrintStream toServer;

String textFromServer;

//socket connection setup to server at port 4141

**try**{

sock= **new** Socket(serverName, 4141);

fromServer=**new** BufferedReader(**new** InputStreamReader(sock.getInputStream()));

toServer=**new** PrintStream(sock.getOutputStream());

toServer.println(name);

// toServer.flush();

//code moved to server during code revision

//Remove all the existing data from jokes

//jokes.clear();

//display joke or proverb sent from the server

textFromServer=fromServer.readLine();

System.***out***.println(textFromServer);

sock.close();

}**catch**(IOException x){

System.***out***.println("Socket error");

x.printStackTrace();

}

}

//code moved to server during code revision

/\*\*static void removeDuplicates(){

for (String joke : jokesCopy) {

if (jokes.contains(joke)) {

jokes.remove(joke);

}

}

}\*\*/

}

**JokeAdminClient**

**import** java.io.BufferedReader;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.PrintStream;

**import** java.net.Socket;

/\*\*--------------------------------------------------------

1. **@author** Prashant Padmashali **@Date** 27, September 2015

2. Java version used, if not the official version for the class:

**@Filename** JokeAdminClient.java

**@version** 1.8.0\_60

**@IDE\_Version** Version: Neon Milestone 1 (4.6.0M1) Build id: 20150820-1211

3. Precise command-line compilation examples / instructions:

**@set** path="<drive>:\<path>\Java\jdk\bin"

**@set** classpath=%classpath%;.;

javac JokeClientAdmin.java

java JokeClientAdmin

4. Precise examples / instructions to run this program:

> javac JokeServer.java

> java JokeServer

> javac JokeClient.java

> java JokeClient

> java JokeClientAdmin.java

> java JokeClientAdmin

**For JokeAdminClient the inputs which can be given to change the mode are**

**‘Joke’ or ‘proverb’ or ‘maintenance’ or ‘shutdown’ or ‘quit’**

All acceptable commands are displayed on the various consoles.

This runs across machines, in which case you have to pass the IP address of

the server to the clients. For example, if the server is running at

140.192.1.22 then you would type:

> java JokeClient 140.192.1.22

> java JokeClientAdmin 140.192.1.22

5. List of files needed for running the program.

a. checklist.html

b. JokeServer.java

c. JokeClient.java

d. JokeClientAdmin.java

5. Notes:

The Program is a joke Admin Client which is built with the help of InetClient.java also the socket implementation of inetserver and inetclient is used as reference

(http://condor.depaul.edu/elliott/435/hw/programs/inet/InetPDF-C.pdf)the admin can control the server's behavior to Joke,Proverb and Maintenance Mode. In return to

which the server will return the output to the client.

----------------------------------------------------------\*\*/

**public** **class** JokeAdminClient {

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

// @sName- ServerName of the computer.assigned as localhost

String sName;

**if** (args.length<1) sName="localhost";

**else** sName=args[0];

System.***out***.println("Prashant's Admin Client \n");

BufferedReader in =**new** BufferedReader(**new** InputStreamReader(System.***in***));

**try**{

String sMode; // to store the mode input from admin

**do**{

System.***out***.println("Enter server mode.\n (1) Joke {Default Mode} \n (2) Proverb \n (3) Maintenance \n (4) 'quit' to end ");

System.***out***.flush();

sMode=in.readLine();

**if** (sMode.indexOf("quit")<0) //if quit is typed the admin client closes

*modeControl*(sName,sMode);

}**while** (sMode.indexOf("quit")<0);

System.***out***.println("Cancelled by user request");

} **catch** (IOException x) {x.printStackTrace();}

}

/\*\*

\* **@sMode**-servermode

\* this is to establish connection between jokeserver and adminclient

\*/

**public** **static** **void** modeControl(String sName, String sMode) {

Socket sock;

BufferedReader fromServer;

PrintStream toServer;

String textFromServer;

**try**{

sock =**new** Socket(sName,4242);//connecting to server @port 4242 and servername

fromServer=**new** BufferedReader(**new** InputStreamReader(sock.getInputStream()));

toServer=**new** PrintStream(sock.getOutputStream());

toServer.println(sMode);toServer.flush();//sending the server mode to the Server

//for(int i=1;i<=3;i++){

textFromServer=fromServer.readLine();

**if**(textFromServer !=**null**) System.***out***.println(textFromServer);

//}

sock.close();

}**catch** (IOException x) {

System.***out***.println("socket error");

x.printStackTrace();

}

}

}

**Output**

**JokeClient**

Prashant's Joke client

Using Server localhost, port 4141

Please enter your name.

prashant

enter 'quit' to close the application or Hit 'enter' key to proceed

**E.J>prashant, Don't drink and drive. You might hit a bump and spill your drink.**

**enter 'quit' to close the application or Hit 'enter' key to proceed**

**D.J>prashant, If you lend someone $20, and never see that person again; it was probably worth it.**

**enter 'quit' to close the application or Hit 'enter' key to proceed**

**A.J>prashant, Error, no keyboard. Press F1 to continue.**

**enter 'quit' to close the application or Hit 'enter' key to proceed**

**C.J>prashant, Everyone has a photographic memory. Some don't have film.**

**enter 'quit' to close the application or Hit 'enter' key to proceed**

**B.J>prashant, Ever notice how fast Windows runs? Neither did I.**

**enter 'quit' to close the application or Hit 'enter' key to proceed**

-------------------------------------------------------------joke set-1

B.J>prashant, Ever notice how fast Windows runs? Neither did I.

enter 'quit' to close the application or Hit 'enter' key to proceed

D.J>prashant, If you lend someone $20, and never see that person again; it was probably worth it.

enter 'quit' to close the application or Hit 'enter' key to proceed

E.J>prashant, Don't drink and drive. You might hit a bump and spill your drink.

enter 'quit' to close the application or Hit 'enter' key to proceed

C.J>prashant, Everyone has a photographic memory. Some don't have film.

enter 'quit' to close the application or Hit 'enter' key to proceed

A.J>prashant, Error, no keyboard. Press F1 to continue.

enter 'quit' to close the application or Hit 'enter' key to proceed

-------------------------------------------------------------joke set-2

A.P>If you want something done right, prashant you have to do it yourself.

enter 'quit' to close the application or Hit 'enter' key to proceed

C.P>prashant Don't count your chickens before they hatch.

enter 'quit' to close the application or Hit 'enter' key to proceed

B.P>prashant One man's trash is another man's treasure.

enter 'quit' to close the application or Hit 'enter' key to proceed

D.P>prashant There's no such thing as a free lunch.

enter 'quit' to close the application or Hit 'enter' key to proceed

E.P>prashant When the going gets tough, the tough get going.

enter 'quit' to close the application or Hit 'enter' key to proceed

-------------------------------------------------------------proverb set-1

B.P>prashant One man's trash is another man's treasure.

enter 'quit' to close the application or Hit 'enter' key to proceed

C.P>prashant Don't count your chickens before they hatch.

enter 'quit' to close the application or Hit 'enter' key to proceed

D.P>prashant There's no such thing as a free lunch.

enter 'quit' to close the application or Hit 'enter' key to proceed

E.P>prashant When the going gets tough, the tough get going.

enter 'quit' to close the application or Hit 'enter' key to proceed

A.P>If you want something done right, prashant you have to do it yourself.

enter 'quit' to close the application or Hit 'enter' key to proceed

-------------------------------------------------------------proverb set-2

A.J>prashant, Error, no keyboard. Press F1 to continue.

enter 'quit' to close the application or Hit 'enter' key to proceed

D.J>prashant, If you lend someone $20, and never see that person again; it was probably worth it.

enter 'quit' to close the application or Hit 'enter' key to proceed

A.P>If you want something done right, prashant you have to do it yourself.

enter 'quit' to close the application or Hit 'enter' key to proceed

D.P>prashant There's no such thing as a free lunch.

enter 'quit' to close the application or Hit 'enter' key to proceed

E.P>prashant When the going gets tough, the tough get going.

enter 'quit' to close the application or Hit 'enter' key to proceed

-------------------------------------------------2 joke & 3 proverb set-1

C.J>prashant, Everyone has a photographic memory. Some don't have film.

enter 'quit' to close the application or Hit 'enter' key to proceed

B.J>prashant, Ever notice how fast Windows runs? Neither did I.

enter 'quit' to close the application or Hit 'enter' key to proceed

E.J>prashant, Don't drink and drive. You might hit a bump and spill your drink.

enter 'quit' to close the application or Hit 'enter' key to proceed

B.P>prashant One man's trash is another man's treasure.

enter 'quit' to close the application or Hit 'enter' key to proceed

C.P>prashant Don't count your chickens before they hatch.

enter 'quit' to close the application or Hit 'enter' key to proceed

-------------------------------------------------3 joke & 2 proverb set-1

Server is down due to Maintenance

enter 'quit' to close the application or Hit 'enter' key to proceed

Server is down due to Maintenance

enter 'quit' to close the application or Hit 'enter' key to proceed

Server is down due to Maintenance

enter 'quit' to close the application or Hit 'enter' key to proceed

**JokeAdminClient**

Prashant's Admin Client

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

proverb

Mode change has been completed to Proverb

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

joke

Mode change has been completed to Joke

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

proverb

Mode change has been completed to Proverb

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

joke

Mode change has been completed to Joke

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

proverb

Mode change has been completed to Proverb

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

Error code:400 Bad Request

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

maintenance

Mode change has been completed to Maintenance

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

shutdown

Please send the final shutdown request to the Server

Enter server mode.

(1) Joke {Default Mode}

(2) Proverb

(3) Maintenance

(4) 'quit' to end

quit

Cancelled by user request

**JokeServer**

In the admin looper thread

Prashant's Joke Server starting up, listening for client port at 4141

Mode change to PROVERBS has been captured

Mode change to JOKES has been captured

Mode change to PROVERBS has been captured

Mode change to JOKES has been captured

Mode change to PROVERBS has been captured

Mode change to MAINTENANCE has been captured

Admin Worker has captured a shutdown request

</pre>