```
1: /*
                                                                                               (chosenServerString);
    2: * @author Haniel Rameshbabu
                                                                                                  59:
                                                                                                                                               --chosenServerInt:
                                                                                                                                                                        //decrement the va
    3: */
                                                                                               lue to match index
    4:
                                                                                                  60:
                                                                                                                                               response = serverstub.joinServer(servers.g
    5: package cs3524.solutions.mud;
                                                                                               et (chosenServerInt), userstub);
    6:
                                                                                                  61:
                                                                                                                                               if ( response == false ) {System.exit(0);}
    7: import java.rmi.Naming;
                                                                                                  62:
    8: import java.lang.SecurityManager;
                                                                                                  63:
                                                                                                                                       else { // invalid input
    9: import java.rmi.server.UnicastRemoteObject;
                                                                                                  64:
                                                                                                                                               chosenServerString = null;
   10: import java.util.List;
                                                                                                  65:
                                                                                                                                               System.out.println("Invalid choice! Try ag
   11: import java.util.Scanner;
                                                                                               ain."):
                                                                                                  66:
   13: public class Client {
                                                                                                  67:
   14.
                                                                                                  68.
   15:
               public static void main(String args[]) {
                                                                                                  69:
   16:
                        if (args.length < 3) {</pre>
                                                                                                  70:
                                                                                                                               // Main interface
   17:
                                System.err.println( "Usage:\njava Client <registryhostname
                                                                                                  71 •
                                                                                                                               System.out.println( "Type 'help' for controls/commands" );
> <registryport> <callbackport>" );
                                                                                                  72:
                                                                                                                               String input;
   18:
                                                                                                  73:
                                return;
                                                                                                                               while (true) {
                                                                                                                                       input = System.console().readLine("\nEnter command
   19:
                                                                                                  74:
   20:
                                                                                               :\n\n").trim();
   21:
                                                                                                  75:
                        try {
   22:
                                                                                                  76:
                                                                                                                                       // Basic commands sys out
   23:
                                                                                                  77:
                                String hostname = args[0];
                                                                                                                                       if (input.equals("help") ) {
   24:
                                int registryport = Integer.parseInt( args[1] );
                                                                                                  78:
                                                                                                                                               System.out.println( "\nview \nmove \ntake
   25.
                                int callbackport = Integer.parseInt( args[2] );
                                                                                               \nshow inventory \nonline users \nmessage \nexit\n" );
   26:
                                                                                                  79:
   27:
                                System.setProperty( "java.security.policy", "mud.policy" )
                                                                                                  80:
                                                                                                  81:
                                                                                                                                       else if ( input.equals( "view" ) )
   28:
                                System.setSecurityManager( new SecurityManager() );
                                                                                                  82:
                                                                                                                                               //view waht you have around you
   29:
                                                                                                  83:
                                                                                                                                               serverstub.view( userName, "paths" );
   30:
                                // User Instance
                                                                                                  84:
                                                                                                                                               serverstub.view( userName, "things" );
   31:
                                String userName = System.console().readLine("Username: ").
                                                                                                  85.
trim();
                                                                                                  86:
   32:
                                UserImpl user = new UserImpl( userName );
                                                                                                  87:
                                                                                                                                       else if ( input.equals( "move" ) )
   33:
                                UserInterface userstub = (UserInterface)UnicastRemoteObjec
                                                                                                  88:
                                                                                                                                               // move somewhere
t.exportObject( user, callbackport );
                                                                                                  89:
                                                                                                                                               System.out.println( "You can move:\n");
   34:
                                                                                                  90:
                                                                                                                                               serverstub.view( userName, "paths" );
   35:
                                                                                                  91:
                                                                                                                                               input = System.console().readLine( "Where
   36:
                                // Stub
                                                                                               do you want to move?\n" ).trim();
   37:
                                String regURL = "rmi://" + hostname + ":" + registryport +
                                                                                                                                               if (input.equals("north") | input.equals
                                                                                                  92:
 "/MUDServer";
                                                                                               ("east") | input.equals("south") | input.equals("west") )
   38:
                                System.out.println("Looking up " + regURL );
                                                                                                  93:
   39:
                                MUDServerInterface serverstub = (MUDServerInterface) Naming
                                                                                                  94:
                                                                                                                                                       serverstub.moveUser( userName, inp
.lookup( regURL );
                                                                                               ut );
   40:
                                                                                                  95:
   41:
                                                                                                  96:
   42:
                                                                                                  97:
                                List<String> servers = serverstub.listServers();
   43:
                                Integer i = 1;
                                                                                                  98:
                                                                                                                                       else if ( input.equals( "take" ) )
   44:
                                for( String srv : servers )
                                                                                                  99:
                                                                                                                                               // take item around you
   45:
                                                                                                 100:
                                                                                                                                               serverstub.view( userName, "things" );
   46:
                                        System.out.println("("+i+") "+srv);
                                                                                                 101:
                                                                                                                                               input = System.console().readLine( "What w
   47:
                                        ++1:
                                                                                               ould you like to take?\n" ).trim();
   48:
                                                                                                 102:
                                                                                                                                               serverstub.getThing( userName, input );
   49:
                                                                                                 103:
   50.
                                                                                                 104.
                                //choose a server or create your own
   51:
                                String chosenServerString = null;
                                                                                                 105:
                                                                                                                                       else if ( input.equals( "show inventory" ) )
   52:
                                                                                                 106:
                                boolean response = false;
                                                                                                                                               //show your items in inventory
   53:
                                while (chosenServerString == null)
                                                                                                 107:
                                                                                                                                               serverstub.showInventory( userName );
   54:
                                                                                                 108:
   55:
                                        chosenServerString = System.console().readLine("Co
                                                                                                 109:
nnect to server number: ").trim();
                                                                                                 110:
                                                                                                                                       else if ( input.equals( "online users" ) )
   56:
                                        if (Integer.parseInt(chosenServerString) <= server</pre>
                                                                                                 111:
                                                                                                                                               // show all online users on the server
s.size())
                                                                                                 112:
                                                                                                                                               serverstub.listUsers( userName );
   57:
                                                 // you have chosen one of the existing ser
                                                                                                 113:
vers
                                                                                                 114:
   58:
                                                 Integer chosenServerInt = Integer.parseInt
                                                                                                 115:
                                                                                                                                       else if ( input.equals( "message" ) )
```

```
116:
                                               // send a message to online user
 117:
                                               System.out.println( "You can message to:\n
");
 118:
                                               serverstub.listUsers( userName );
  119:
                                               String to = System.console().readLine( "Wr
ite the name:\n" ).trim();
                                               String message = System.console().readLine
( "Write the message: \n" ).trim();
 121:
                                               serverstub.message(userName, to, message)
  122:
  123:
  124:
  125:
 126:
                                       // Exit
 127:
                                       else if ( input.equals( "exit" ) ) {
  128:
                                               //serverStub.leaveServer( userName );
 129:
                                               System.exit(0);
 130:
 131:
                                       // Invalid command
 132:
 133:
                                       else {
 134:
                                               System.out.println("Please enter valid com
mand.\n");
 135:
 136:
 137:
 138:
 139:
 140:
                       catch(java.rmi.NotBoundException e) {
 141:
                               //System.err.println( "Can't find the auctioneer in the re
gistry.");
 142:
                               System.err.println("Error: server NotBoundException");
 143:
  144:
                       catch (java.io.IOException e) {
  145:
                               System.out.println( "Failed to register." );
  146:
  147:
  148:
```

Mon Apr 01 04:18:08 2019

2

Client.java

149: }

```
Edge.java Fri
```

```
Fri Mar 22 19:29:21 2019
```

```
1
```

```
1: /**********************
2: * cs3524.solutions.mud.Edge
5: package cs3524.solutions.mud;
7: // Represents an path in the MUD (an edge in a graph).
8: class Edge
9: {
      public Vertex _dest; // Your destination if you walk down this path
10:
      public String _view; // What you see if you look down this path
11:
12:
13:
      public Edge( Vertex d, String v )
14:
15:
         _{dest} = d;
16:
               _{view} = v;
17:
18: }
19:
```

```
2: * cs3524.solutions.mud.MUD
 5: package cs3524.solutions.mud;
 6:
7: import java.io.FileReader;
8: import java.io.BufferedReader;
9: import java.io.IOException;
10: import java.util.StringTokenizer;
12: import java.util.Iterator;
13: import java.util.List;
14: import java.util.Map;
15: import java.util.Vector;
16: import java.util.HashMap;
18: /**
19: * A class that can be used to represent a MUD; essenially, this is a
20: * graph.
21: */
23: public class MUD
24: {
25:
26:
        * Private stuff
27:
28:
29:
       // A record of all the vertices in the MUD graph. HashMaps are not
30:
       // synchronized, but we don't really need this to be synchronised.
31:
       private Map<String,Vertex> vertexMap = new HashMap<String,Vertex>();
32:
33:
       private String startLocation = "";
34:
35:
36:
        * Add a new edge to the graph.
37:
38:
       private void addEdge ( String sourceName,
39:
                           String destName,
40:
                           String direction,
41:
                           String view )
42:
43:
           Vertex v = getOrCreateVertex( sourceName );
44:
           Vertex w = getOrCreateVertex( destName );
45:
           v. routes.put(direction, new Edge(w, view));
46:
47:
48:
49:
        * Create a new thing at a location.
50:
51:
       private void createThing( String loc,
52:
                               String thing )
53:
54:
               Vertex v = getOrCreateVertex( loc );
55:
                v._things.add( thing );
56:
57:
58:
       /**
59:
        * Change the message associated with a location.
60:
61:
       private void changeMessage( String loc, String msg )
62:
63:
               Vertex v = getOrCreateVertex( loc );
64:
                v.\_msq = msq;
65:
66:
       /**
67:
```

```
68:
          * If vertexName is not present, add it to vertexMap. In either
 69.
          * case, return the Vertex. Used only for creating the MUD.
70:
71:
        private Vertex getOrCreateVertex( String vertexName )
72.
73:
             Vertex v = vertexMap.get( vertexName );
74:
             if (v == null) {
75:
                 v = new Vertex( vertexName );
76:
                 vertexMap.put( vertexName, v );
77:
78:
             return v;
79:
        }
80:
81:
         /**
82:
          */
83:
84:
        private Vertex getVertex( String vertexName )
85:
86:
                  return vertexMap.get( vertexName );
87:
88:
89.
90:
          * Creates the edges of the graph on the basis of a file with the
91:
          * following fromat:
92:
          * source direction destination message
93:
94:
         private void createEdges( String edgesfile )
95:
96:
             try {
97:
                 FileReader fin = new FileReader( edgesfile );
98:
                     BufferedReader edges = new BufferedReader(fin);
99:
                     String line;
100:
                     while((line = edges.readLine()) != null) {
101:
                         StringTokenizer st = new StringTokenizer( line );
102:
                     if( st.countTokens() < 3 ) {</pre>
103:
                         System.err.println( "Skipping ill-formatted line " + line );
104:
                         continue;
105:
106:
                     String source = st.nextToken();
107:
                     String dir = st.nextToken();
108:
                     String dest = st.nextToken();
109:
                     String msg = "";
110:
                     while (st.hasMoreTokens()) {
111:
                         msg = msg + st.nextToken() + " ";
112:
113:
                     addEdge( source, dest, dir, msg );
114:
115:
116:
             catch( IOException e ) {
117:
                 System.err.println( "Graph.createEdges( String " +
118:
                                     edgesfile + ")\n" + e.getMessage() );
119:
120:
        }
121:
122:
123:
          {}^{\star} Records the messages assocated with vertices in the graph on
124:
          * the basis of a file with the following format:
125:
          * location message
126:
          * The first location is assumed to be the starting point for
127:
          * users joining the MUD.
128:
129:
         private void recordMessages( String messagesfile )
130:
131:
132:
                 FileReader fin = new FileReader( messagesfile );
133:
                     BufferedReader messages = new BufferedReader( fin );
134:
                     String line;
```

```
135:
                  boolean first = true; // For recording the start location.
                                                                                               202:
                                                                                                            recordMessages( messagesfile );
                                                                                               203:
136.
                      while((line = messages.readLine()) != null) {
                                                                                                            recordThings( thingsfile );
137:
                          StringTokenizer st = new StringTokenizer( line );
                                                                                               204:
138:
                      if( st.countTokens() < 2 ) {</pre>
                                                                                               205:
                                                                                                            System.out.println( "Files read..." );
139.
                          System.err.println( "Skipping ill-formatted line " + line );
                                                                                               206.
                                                                                                            System.out.println( vertexMap.size() + " vertices\n" );
140:
                          continue;
                                                                                               207:
141:
                                                                                               208:
142:
                      String loc = st.nextToken();
                                                                                               209.
                                                                                                            // This method enables us to display the entire MUD (mostly used
143:
                      String msg = "";
                                                                                               210:
                                                                                                            // for testing purposes so that we can check that the structure
                      while (st.hasMoreTokens()) {
                                                                                               211:
                                                                                                            // defined has been successfully parsed.
144:
145:
                                                                                               212:
                                                                                                            public String toString()
                          msg = msg + st.nextToken() + " ";
146:
                                                                                               213:
147:
                      changeMessage (loc, msg);
                                                                                               214:
                                                                                                            String summary = "";
148 •
                      if (first) {
                                        // Record the start location.
                                                                                               215:
                                                                                                            Iterator iter = vertexMap.keySet().iterator();
149:
                          startLocation = loc;
                                                                                               216:
                                                                                                            String loc:
150:
                          first = false:
                                                                                               217:
                                                                                                            while (iter.hasNext()) {
151:
                                                                                               218.
                                                                                                                loc = (String)iter.next();
152:
                                                                                               219:
                                                                                                                summary = summary + "Node: " + loc;
153:
                                                                                               220:
                                                                                                                summary += ((Vertex)vertexMap.get(loc)).toString();
154:
             catch( IOException e ) {
                                                                                               221:
155:
                 System.err.println( "Graph.recordMessages( String " +
                                                                                               222:
                                                                                                            summary += "Start location = " + _startLocation;
                                      messagesfile + ")\n" + e.getMessage() );
                                                                                               223:
156:
                                                                                                            return summary;
157:
                                                                                               224:
                                                                                               225:
158:
159:
                                                                                               226:
                                                                                               227:
160:
                                                                                                         * A method to provide a string describing a particular location.
161:
          * Records the things assocated with vertices in the graph on
                                                                                               228:
162:
           * the basis of a file with the following format:
                                                                                               229:
                                                                                                        public String locationInfo( String loc )
           * location thing1 thing2 ...
                                                                                               230:
163:
                                                                                               231:
164:
                                                                                                                 return getVertex( loc ).toString();
165:
         private void recordThings( String thingsfile )
                                                                                               232:
                                                                                               233:
166:
                                                                                               234:
167:
                                                                                                         * Get the start location for new MUD users.
168:
                  FileReader fin = new FileReader (thingsfile);
                                                                                               235:
169:
                      BufferedReader things = new BufferedReader( fin );
                                                                                               236:
170:
                      String line;
                                                                                               237:
                                                                                                        public String startLocation()
171:
                      while((line = things.readLine()) != null) {
                                                                                               238:
172:
                          StringTokenizer st = new StringTokenizer( line );
                                                                                               239:
                                                                                                                 return startLocation;
173:
                                                                                               240:
                      if( st.countTokens() < 2 ) {</pre>
174:
                          System.err.println( "Skipping ill-formatted line " + line );
                                                                                               241:
175:
                                                                                               242:
                          continue;
176:
                                                                                               243:
177:
                      String loc = st.nextToken();
                                                                                               244:
                                                                                                         * method that provides infor where a player can move
178:
                                                                                               245:
                      while (st.hasMoreTokens()) {
179:
                                                                                               246:
                          addThing( loc, st.nextToken());
                                                                                                        public String locationPaths( String loc )
180:
                                                                                               247:
181:
                                                                                               248:
                                                                                                            String message = getVertex( loc ). msg + "\n";
182:
                                                                                               249:
                                                                                                            for (Map.Entry<String, Edge> vertex : getVertex(loc)._routes.entrySet())
183:
             catch( IOException e ) {
                                                                                               250:
184:
                  System.err.println( "Graph.recordThings( String " +
                                                                                               251:
                                                                                                                message += "You can move to the " + vertex.getKey() + " there is " + v
185 •
                                      thingsfile + ") \n" + e.getMessage() );
                                                                                            ertex.getValue()._view + "\n";
186:
                                                                                               252:
187:
                                                                                               253:
                                                                                                            return message;
188:
                                                                                               254:
189:
                                                                                               255:
190:
          ^{\star} All the public stuff. These methods are designed to hide the
                                                                                               256:
                                                                                                        //method that provides info about things on location
191:
           ^{\star} internal structure of the MUD. Could declare these on an
                                                                                               257:
                                                                                                        public List locationThings( String loc )
192:
           * interface and have external objects interact with the MUD via
                                                                                               258:
                                                                                               259:
193:
           * the interface.
                                                                                                            List<String> things = getVertex(loc)._things;
194:
                                                                                               260:
                                                                                                            return things;
                                                                                               261:
195:
                                                                                                        }
         /**
196:
                                                                                               262:
197:
          * A constructor that creates the MUD.
                                                                                               263:
198:
                                                                                               264:
                                                                                                         * Add a thing to a location; used to enable us to add new users.
199:
         public MUD( String edgesfile, String messagesfile, String thingsfile )
                                                                                               265:
200:
                                                                                               266:
                                                                                                        public void addThing( String loc,
201:
             createEdges( edgesfile );
                                                                                               267:
                                                                                                                               String thing )
```

```
MUD.java
                      Mon Apr 01 02:10:05 2019
  268:
  269:
               Vertex v = getVertex( loc );
  270:
               v._things.add( thing );
  271:
  272:
  273:
            * Remove a thing from a location.
  274:
  275:
           public void delThing( String loc,
  276:
  277:
                                 String thing )
  278:
  279:
               Vertex v = getVertex( loc );
  280:
               v._things.remove( thing );
  281:
  282:
  283:
  284:
            * A method to enable a player to move through the MUD (a player
  285:
            * is a thing). Checks that there is a route to travel on. Returns
  286:
            * the location moved to.
  287:
            */
  288:
           public String moveThing( String loc, String dir, String thing )
  289:
  290:
               Vertex v = getVertex( loc );
  291:
               Edge e = v._routes.get( dir );
  292:
               if (e == null) // if there is no route in that direction
  293:
                   return loc; // no move is made; return current location.
  294:
               v._things.remove( thing );
  295:
               e._dest._things.add( thing );
  296:
               return e._dest._name;
  297:
  298:
           /**
  299:
            * A main method that can be used to testing purposes to ensure
  300:
  301:
            * that the MUD is specified correctly.
  302:
  303:
           public static void main(String[] args)
  304:
  305:
               if (args.length != 3) {
  306:
                   System.err.println("Usage: java Graph <edgesfile> <messagesfile> <thin
gsfile>");
  307:
                   return;
  308:
  309:
               MUD m = new MUD( args[0], args[1], args[2] );
```

System.out.println(m.toString());

310:

311: 312: }

```
1: /*
                                                                                             ception {
    2: * @author Haniel Rameshbabu
                                                                                                63:
                                                                                                        MUD server = servers.get(servername);
    3: * Implementation of MUDServerInterface
                                                                                                64:
                                                                                                        String userName = user.getName();
    4: */
                                                                                                65:
                                                                                                        HashMap<String, UserInterface> userObj;
    5.
                                                                                                66:
    6: package cs3524.solutions.mud;
                                                                                                67:
                                                                                                68:
                                                                                                        /*if(loggedUsers.get(name).size() >= maxPlayers) { // check if maxPlayers is
    8: import java.rmi.Remote;
                                                                                             reached
    9: import java.rmi.RemoteException;
                                                                                                69:
                                                                                                          client.sendMessage( "Sorry maximum players reached. Try later or join anothe
   10: import java.util.Iterator;
                                                                                             r server" );
   11: import java.util.List;
                                                                                                70:
                                                                                                          return false;
   12: import java.util.Map;
                                                                                                71:
   13: import java.util.ArravList;
                                                                                                72:
   14: import java.util.Set;
                                                                                                73:
                                                                                                        if (userMap.get(userName) != null) { // check if userName already exist
   15: import java.util.HashMap;
                                                                                             . It must be unique
   16:
                                                                                                74:
                                                                                                          client.sendMessage( "Change name please! User already exist!" );
   17:
                                                                                                75:
                                                                                                          return false:
   18: public class MUDServerImpl implements MUDServerInterface {
                                                                                                76:
                                                                                                        1*/
                                                                                                77:
   19:
   20:
         // Key => Value; name => MUD object
                                                                                                78:
                                                                                                        // User
   21:
         private Map<String, MUD> servers = new HashMap<String, MUD>();
                                                                                                79:
                                                                                                        userMap.put ( userName, servername );
                                                                                                        userObj = loggedUsers.get( servername );
   22:
                                                                                                80:
   23.
                                                                                                81:
         // whereabouts of the User
   24:
                                                                                                82:
                                                                                                        userObj.put(userName, user);
               private Map<String, String> userMap = new HashMap<String, String>();
   25:
         // records of all users positions
                                                                                                83:
   26:
                                                                                                84:
               private Map<String, String> userPosMap = new HashMap<String, String>();
                                                                                                        loggedUsers.put( servername, userObj );
   27:
                                                                                                85:
   28:
                                                                                                86:
               // server => Hash of all usernames logged
                                                                                                        userPosMap.put( userName, server.startLocation() );
   29:
               private Map<String, HashMap<String, UserInterface>> loggedUsers = new Hash
                                                                                                87:
Map<String, HashMap<String, UserInterface>>();
                                                                                                88:
                                                                                                        inventoryMap.put( userName, new ArrayList<String>() );
   30:
                                                                                                89:
   31:
                                                                                                90:
               // holds records of all user inventories
                                                                                                        server.addThing( server.startLocation(), "User: "+userName );
   32:
         // @format InventpryName => InventoryItem
                                                                                                91:
   33:
               private Map<String, ArrayList<String>> inventoryMap = new HashMap<String,</pre>
                                                                                                92:
ArrayList<String>>();
                                                                                                93:
                                                                                                        // prepare the message
   34:
                                                                                                94:
                                                                                                        String message = ( "\n**** Welcome to " + servername + " Server ****\n" );
   35:
         // Create all servers at start; i.e 3;
                                                                                                95:
                                                                                                        message += "Current number of player on this server is "+loggedUsers.get( serv
   36:
         public MUDServerImpl() throws RemoteException {
                                                                                             ername ).size()+"\n";
   37:
           // Server 1 (Rosa)
                                                                                                96:
                                                                                                        message += "You are currently at "+userPosMap.get( userName )+" location\n";
   38:
           servers.put("Rosa", new MUD("MUDs/rosa/rosa.edg", "MUDs/rosa/rosa.msg", "MUDs/ro
                                                                                                97:
                                                                                                98:
sa/rosa.thq"));
                                                                                                        // send the message to the client
   39:
           // Username => Object
                                                                                                99:
                                                                                                        user.sendMessage ( message );
   40:
           HashMap<String, UserInterface> RosaObj = new HashMap<String, UserInterface>();
                                                                                               100:
                                                                                                        return true;
   41:
                       loggedUsers.put( "Rosa", RosaObj);
                                                                                               101:
   42:
                                                                                               102:
   43:
           // Server 2
                                                                                               103:
                                                                                                      // view what is at particular location
   44:
           servers.put("Tros", new MUD("MUDs/tros/tros.edq", "MUDs/tros/tros.msq", "MUDs/tr
                                                                                               104:
                                                                                                      public boolean view( String userName, String what ) throws RemoteException{
                                                                                               105:
os/tros.thq"));
                                                                                                        String serverName = userMap.get( userName );
                                                                                                        MUD server = servers.get( serverName );
   45:
           // Username => Object
                                                                                               106:
   46:
           HashMap<String, UserInterface> TrosObj = new HashMap<String, UserInterface>();
                                                                                               107:
                                                                                                        HashMap<String, UserInterface> clientsMap = loggedUsers.get( serverName );
   47:
           loggedUsers.put( "Tros", TrosObj);
                                                                                               108:
                                                                                                        UserInterface client = clientsMap.get( userName );
   48:
                                                                                               109:
                                                                                                        String position = userPosMap.get( userName );
   49:
           // Server 3
                                                                                               110:
                                                                                                        String message = null;
   50:
           servers.put("Gina", new MUD("MUDs/gina/gina.edg", "MUDs/gina/gina.msg", "MUDs/gi
                                                                                               111:
na/gina.thg"));
                                                                                               112:
                                                                                                        if ( what.equals("paths") )
           // Username => Object
                                                                                               113:
   51:
                                                                                                        { // send info about possible paths
   52:
           HashMap<String, UserInterface> GinaObj = new HashMap<String, UserInterface>();
                                                                                               114:
                                                                                                          message = server.locationPaths( position );
   53:
                                                                                               115:
           loggedUsers.put( "Gina", GinaObj);
                                                                                                          client.sendMessage( message );
   54:
                                                                                               116:
                                                                                                          return true;
                                                                                               117:
   55:
                                                                                               118:
   56:
   57:
         public List<String> listServers() throws RemoteException {
                                                                                               119:
                                                                                                        if ( what.equals("things") )
   58:
           Set<String> set = servers.keySet();
                                                                                               120:
                                                                                                        { // send info about things
   59:
                       return new ArrayList<String>(set);
                                                                                               121:
                                                                                                          message = "There is:\n";
   60:
                                                                                               122:
                                                                                                          List<String> things = server.locationThings( position );
   61:
                                                                                               123:
                                                                                                          for ( String t : things )
         public boolean joinServer(String servername, UserInterface user) throws RemoteEx
                                                                                               124:
                                                                                                          { // construct the message and send
```

```
125:
             message += t + "\n";
126:
127:
           client.sendMessage( message );
128:
           return true;
129.
130:
         return false:
131: }
132:
133:
       public boolean moveUser(String userName, String position) throws RemoteException
134 •
       { // move the user
         String serverName = userMap.get( userName );
135:
136.
         MUD server = servers.get( serverName );
137:
         HashMap<String, UserInterface> clientsMap = loggedUsers.get( serverName );
138.
         UserInterface client = clientsMap.get( userName );
139:
         String origin = userPosMap.get(userName);
140:
         String message = "";
141:
         // try to move the user and check response
142:
         message = server.moveThing( origin, position, "User: "+userName );
143:
         userPosMap.put ( userName, message );
144:
         if ( message.equals( origin ) )
145:
         { // user is at the same place because there is no path
146:
           client.sendMessage( "You cannot move there.\n" );
147:
           return false;
148:
149:
         client.sendMessage( "You moved to " + message + "\n");
150:
         return true:
151:
152:
153:
154:
155:
       public boolean getThing( String userName, String thing) throws RemoteException
156:
       { // client can take a thing but not a user
157:
158:
         String serverName = userMap.get( userName );
159:
         MUD server = servers.get( serverName );
160:
         HashMap<String, UserInterface> clientsMap = loggedUsers.get( serverName );
161:
         UserInterface client = clientsMap.get( userName );
162:
         ArrayList<String> inventory = inventoryMap.get( userName );
163:
         List<String> things = server.locationThings( userPosMap.get( userName ) );
164:
165:
         for ( String t : things )
166:
         { // iterate through things
167:
           if ( thing.equals( t ) && !thing.contains("User:") )
168:
           { // check if there is the thing client wants to take
169:
             // && check of the thing is not user
170:
             server.delThing( userPosMap.get( userName ), t );
171:
             inventory.add(t);
172:
             inventoryMap.put( userName, inventory );
173:
             client.sendMessage( "You have: "+inventory.toString() );
174:
             return true;
175:
176:
177:
         // the thing was not there or it was a user
178:
         client.sendMessage( "No!\nYou have: "+inventory.toString() );
179:
         return false;
180:
181:
182:
183:
       public boolean showInventory ( String userName ) throws RemoteException
184:
       { // list all the collected items
185:
         String serverName = userMap.get( userName );
186:
         MUD server = servers.get( serverName );
187:
         HashMap<String, UserInterface> clientsMap = loggedUsers.get( serverName );
188:
         UserInterface client = clientsMap.get( userName );
189:
         List<String> inventory = inventoryMap.get(userName);
190:
         String message = "In your inventory is:\n";
191:
         client.sendMessage( message+inventory.toString() );
```

```
192:
          return true;
  193: }
  194 •
  195 •
        public boolean listUsers ( String userName ) throws RemoteException
  196.
        { // list all the online users at the server where client is.
 197:
          String serverName = userMap.get( userName );
  198:
          MUD server = servers.get ( serverName );
  199:
          HashMap<String, UserInterface> clientsMap = loggedUsers.get( serverName );
  200:
          UserInterface client = clientsMap.get( userName );
  201:
          String message = "\nThese users are online:\n";
  202:
           Set<String> clientsSet = clientsMap.keySet();
  203:
           for (String c : clientsSet )
  204:
  205:
            message += c+"\n";
  206:
  207:
          client.sendMessage( message );
  208:
          return true:
  209: }
 210:
  211: public boolean message ( String userName, String to, String message ) throws Remo
teException
 212: { // send a message to a user
 213:
          String serverName = userMap.get( userName );
 214:
          MUD server = servers.get( serverName );
 215:
          HashMap<String, UserInterface> clientsMap = loggedUsers.get( serverName );
 216:
          UserInterface fromClient = clientsMap.get( userName );
 217:
          UserInterface toClient = clientsMap.get( to );
 218:
          String formatedMessage = "Message from " + userName + ":\n" + message;
 219:
          toClient.sendMessage(formatedMessage);
  220:
          fromClient.sendMessage( "\nMessage sent\n");
 221:
          return true;
  222: }
  223:
 224:
  225: }
```

```
1: /*
    2: * @author Haniel Rameshbabu
    3: * Remote Interface for MUD game server
    4: */
    5:
    6: package cs3524.solutions.mud;
    8: import java.rmi.Remote;
    9: import java.rmi.RemoteException;
   10: import java.util.List;
   13: public interface MUDServerInterface extends Remote
   14: {
   15:
   16:
               public List<String> listServers() throws RemoteException;
   17:
               public boolean joinServer( String servername, UserInterface user ) throws
RemoteException;
   18:
               //public boolean leaveServer( String userName ) throws RemoteException;
   19:
               public boolean view( String userName, String way ) throws RemoteException;
   20:
               public boolean moveUser( String userName, String position ) throws RemoteE
xception;
               public boolean getThing( String userName, String thing ) throws RemoteExce
   21:
ption;
   22:
               public boolean showInventory( String userName ) throws RemoteException;
   23:
               public boolean listUsers( String userName ) throws RemoteException;
   24:
               public boolean message( String userName, String to, String message ) throw
s RemoteException;
   25: }
```

```
1: /*
    2: * @author Haniel Rameshbabu
    3: * @title Server Mainline for MUD game server
    4: * @source Adapted Practicals 1 code
    5: */
    7: package cs3524.solutions.mud;
    8:
    9: import java.io.InputStreamReader;
   10: import java.net.InetAddress;
   11: import java.rmi.Naming;
   12: import java.rmi.RMISecurityManager;
   13: import java.rmi.server.UnicastRemoteObject;
   15: public class MUDServerMainline {
   16:
   17:
   18:
   19:
               public static void main(String args[]){
   20:
   21:
                       if (args.length < 2) {</pre>
   22:
                               System.err.println("Usage: \njava MUDServerMainline <regis
tryport> <serverport>");
   23:
                               return;
   24:
   25:
   26:
                       try {
   27:
                               String hostname = (InetAddress.getLocalHost()).getCanonica
lHostName();
   28:
                               int registryport = Integer.parseInt(args[0]);
   29:
                               int serverport = Integer.parseInt(args[1]);
   30:
   31:
                               // Security Policy
   32:
                               System.setProperty("java.security.policy", "mud.policy");
   33:
                               System.setSecurityManager( new SecurityManager());
   34:
                               // Generate remote Objects
   35:
   36:
                               MUDServerImpl server = new MUDServerImpl();
   37:
                               MUDServerInterface stub = (MUDServerInterface) UnicastRemot
eObject.exportObject(server, serverport);
   38:
   39:
                               String regURL = "rmi://" + hostname + ":" + registryport +
 "/MUDServer";
   40:
                               System.out.println("Registering " + regURL);
                               Naming.rebind(regURL, stub);
   41:
   42:
   43:
   44:
   45:
                       // Error Catching
   46:
                       catch(java.net.UnknownHostException e) {
   47:
                           System.err.println( "Cannot determine localhost name." );
   48:
                           System.err.println( e.getMessage() );
   49:
   50:
                       catch (java.io.IOException e) {
   51:
                   System.err.println( "Failed to register!" );
   52:
                           System.err.println( e.getMessage() );
   53:
   54:
   55:
   56:
   57:
   58: }
```

```
1: /*
2: * @author Haniel Rameshbabu
3: */
5: package cs3524.solutions.mud;
7: //import java.util.List;
8: //import java.util.ArrayList;
10: public class UserImpl implements UserInterface {
11:
           private String userName;
12:
           public UserImpl( String name ) {
13:
14:
                   userName = name;
15:
16:
17:
           public String getName() {
18:
                        return userName;
19:
20:
           public void sendMessage(String message) {
21:
22:
                   System.out.println(message);
23:
24: }
```

```
1: /*************************
   2: * cs3524.solutions.mud.Vertex
   5: package cs3524.solutions.mud;
   6:
   7: import java.util.Map;
   8: import java.util.HashMap;
   9: import java.util.List;
  10: import java.util.Vector;
  11: import java.util.Iterator;
  13: // Represents a location in the MUD (a vertex in the graph).
  14: class Vertex
  15: {
          public String _name;
                                        // Vertex name
  16:
  17:
          public String _msg = "";
                                        // Message about this location
  18:
          public Map<String, Edge> _routes; // Association between direction
                                        // (e.g. "north") and a path
  19:
  20:
                                        // (Edge)
  21:
          public List<String> _things;
                                        // The things (e.g. players) at
  22:
                                        // this location
  23:
  24:
          public Vertex( String nm )
  25:
  26:
              _name = nm;
  27:
              _routes = new HashMap<String, Edge>(); // Not synchronised
  28:
              _things = new Vector<String>();
                                                // Synchronised
  29:
  30:
  31:
          public String toString()
  32:
  33:
              String summary = "\n";
  34:
              summary += _msq + "\n";
  35:
              Iterator iter = _routes.keySet().iterator();
  36:
              String direction;
  37:
              while (iter.hasNext()) {
  38:
                 direction = (String)iter.next();
  39:
                 summary += "To the " + direction + " there is " + ((Edge)_routes.get(
direction ))._view + "\n";
  40:
  41:
              iter = things.iterator();
  42:
              if (iter.hasNext()) {
  43:
                 summary += "You can see: ";
  44:
  45:
                     summary += iter.next() + " ";
  46:
                 } while (iter.hasNext());
  47:
              summary += "\n^*;
  48:
              return summary;
  49:
  50:
  51: }
  52:
```