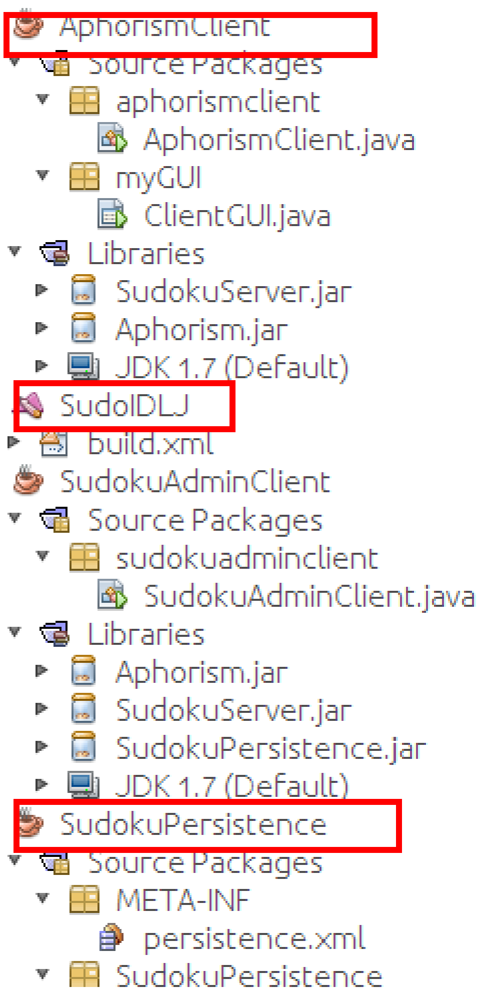
CSCI398 Assignment 2

Dan XIE

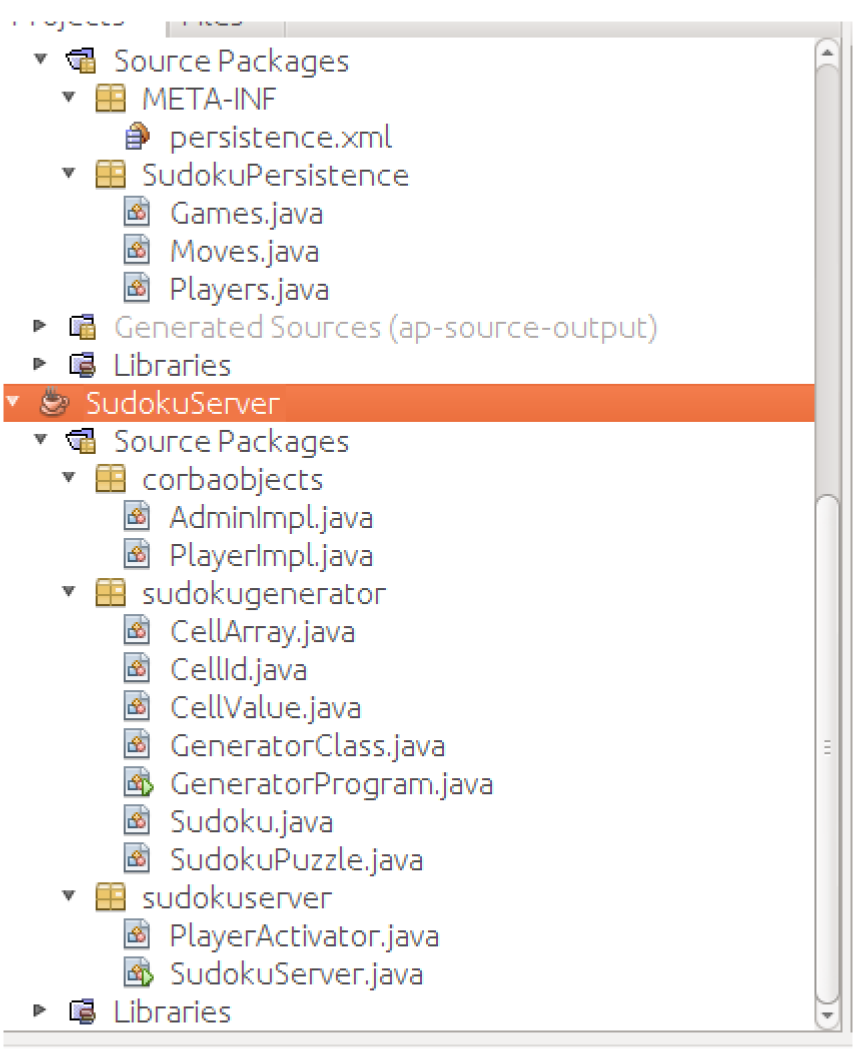
Dx869

4208869

Overview

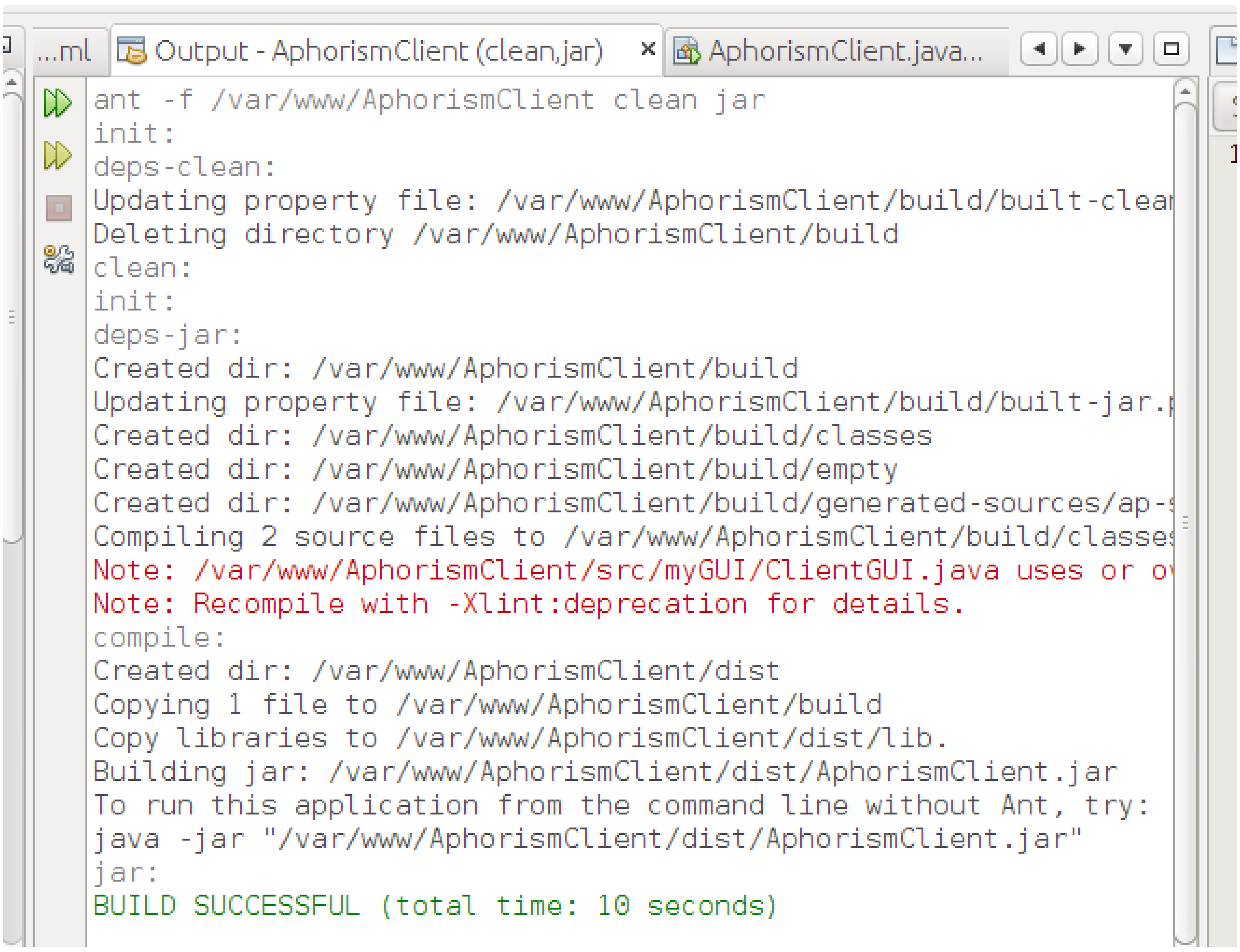


```
AphorismClient
▼ Source Packages
  ▼ aphorismclient
      AphorismClient.java
  ▼ myGUI
      ClientGUI.java
▼ Libraries
  ▶ SudokuServer.jar
  ▶ Aphorism.jar
  ▶ JDK 1.7 (Default)
SudoIDLJ
▶ build.xml
SudokuAdminClient
▼ Source Packages
  ▼ sudokuadminclient
      SudokuAdminClient.java
▼ Libraries
  ▶ Aphorism.jar
  ▶ SudokuServer.jar
  ▶ SudokuPersistence.jar
  ▶ JDK 1.7 (Default)
SudokuPersistence
▼ Source Packages
  ▼ META-INF
      persistence.xml
  ▼ SudokuPersistence
```

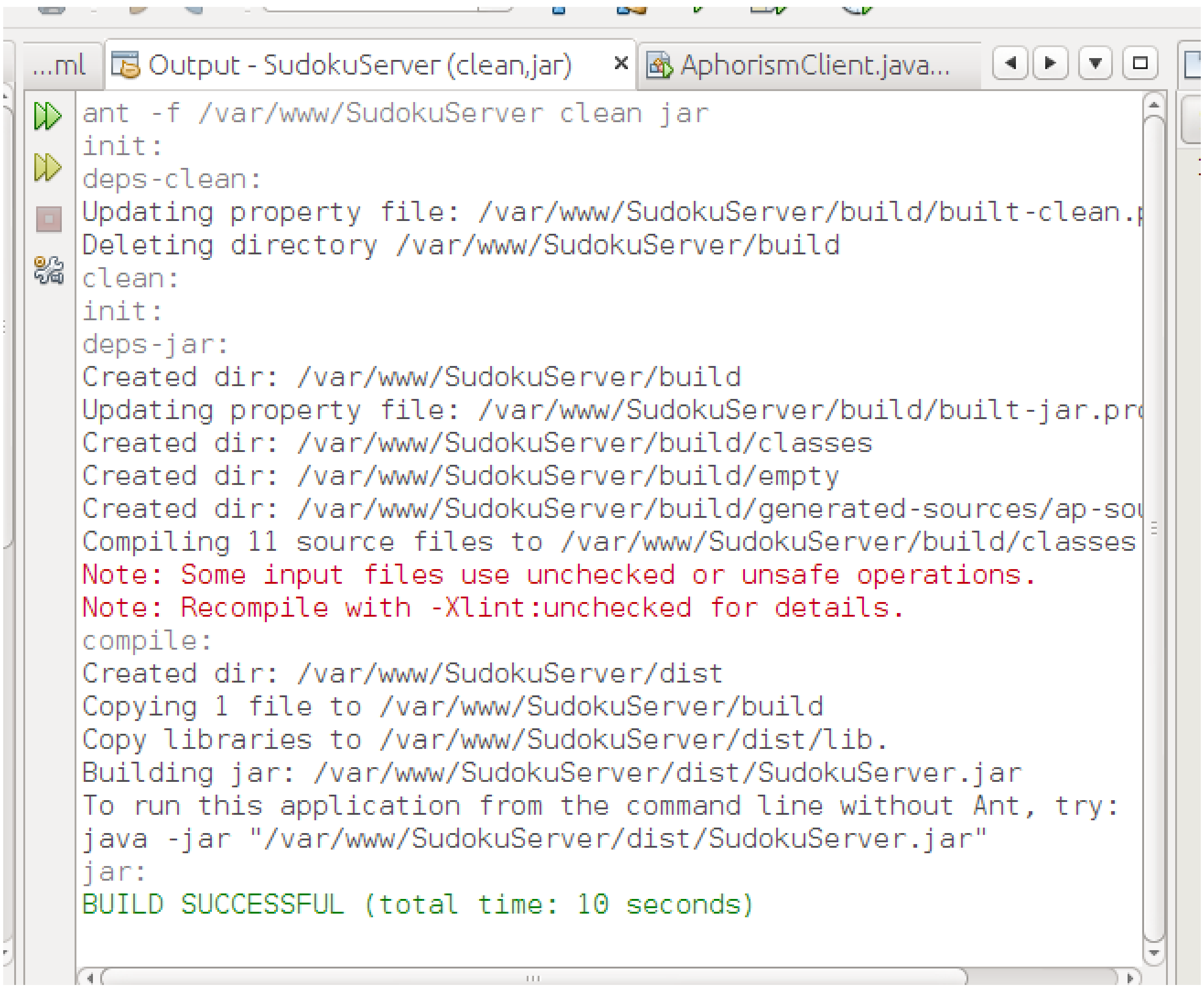


```
▼ Source Packages
  ▼ META-INF
      persistence.xml
  ▼ SudokuPersistence
      Games.java
      Moves.java
      Players.java
▶ Generated Sources (ap-source-output)
▶ Libraries
▼ SudokuServer
  ▼ Source Packages
    ▼ corbaobjects
        AdminImpl.java
        PlayerImpl.java
    ▼ sudokugenerator
        CellArray.java
        CellId.java
        CellValue.java
        GeneratorClass.java
        GeneratorProgram.java
        Sudoku.java
        SudokuPuzzle.java
    ▼ sudokuserver
        PlayerActivator.java
        SudokuServer.java
  ▶ Libraries
```

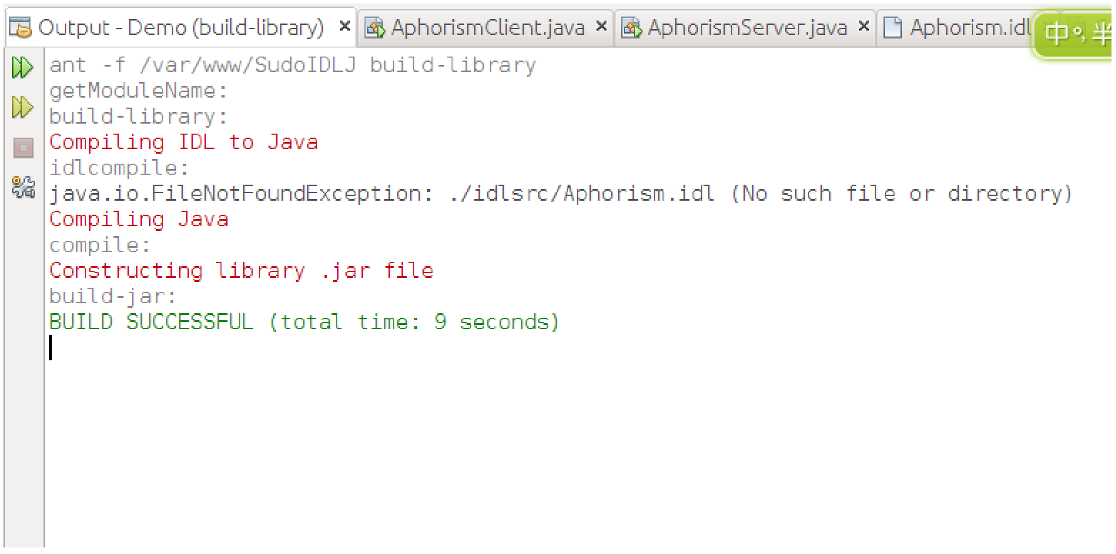
Compiling



```
ant -f /var/www/AphorismClient clean jar
init:
deps-clean:
Updating property file: /var/www/AphorismClient/build/built-clean
Deleting directory /var/www/AphorismClient/build
clean:
init:
deps-jar:
Created dir: /var/www/AphorismClient/build
Updating property file: /var/www/AphorismClient/build/built-jar.p
Created dir: /var/www/AphorismClient/build/classes
Created dir: /var/www/AphorismClient/build/empty
Created dir: /var/www/AphorismClient/build/generated-sources/ap-s
Compiling 2 source files to /var/www/AphorismClient/build/classes
Note: /var/www/AphorismClient/src/myGUI/ClientGUI.java uses or ov
Note: Recompile with -Xlint:deprecation for details.
compile:
Created dir: /var/www/AphorismClient/dist
Copying 1 file to /var/www/AphorismClient/build
Copy libraries to /var/www/AphorismClient/dist/lib.
Building jar: /var/www/AphorismClient/dist/AphorismClient.jar
To run this application from the command line without Ant, try:
java -jar "/var/www/AphorismClient/dist/AphorismClient.jar"
jar:
BUILD SUCCESSFUL (total time: 10 seconds)
```



```
ant -f /var/www/SudokuServer clean jar
init:
deps-clean:
Updating property file: /var/www/SudokuServer/build/built-clean.p
Deleting directory /var/www/SudokuServer/build
clean:
init:
deps-jar:
Created dir: /var/www/SudokuServer/build
Updating property file: /var/www/SudokuServer/build/built-jar.pro
Created dir: /var/www/SudokuServer/build/classes
Created dir: /var/www/SudokuServer/build/empty
Created dir: /var/www/SudokuServer/build/generated-sources/ap-sou
Compiling 11 source files to /var/www/SudokuServer/build/classes
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
compile:
Created dir: /var/www/SudokuServer/dist
Copying 1 file to /var/www/SudokuServer/build
Copy libraries to /var/www/SudokuServer/dist/lib.
Building jar: /var/www/SudokuServer/dist/SudokuServer.jar
To run this application from the command line without Ant, try:
java -jar "/var/www/SudokuServer/dist/SudokuServer.jar"
jar:
BUILD SUCCESSFUL (total time: 10 seconds)
```

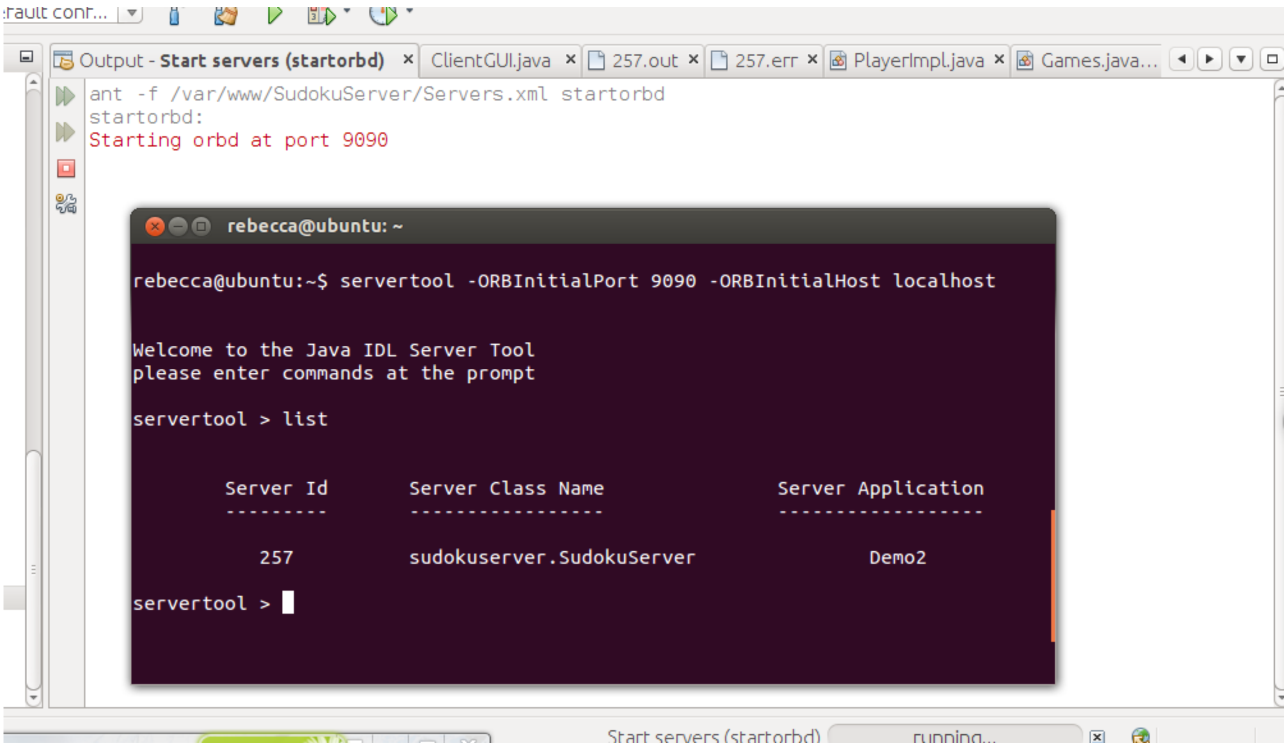


```
ant -f /var/www/SudoIDLJ build-library
getModuleName:
build-library:
Compiling IDL to Java
idlcompile:
java.io.FileNotFoundException: ./idlsrc/Aphorism.idl (No such file or directory)
Compiling Java
compile:
Constructing library .jar file
build-jar:
BUILD SUCCESSFUL (total time: 9 seconds)
```

Code & Display

1. Server for all

Orbd is running at port 9090 and using servertool at port 9090 there is a server registered here waiting for client to invoke it.



```
ant -f /var/www/SudokuServer/Servers.xml startorbd
startorbd:
Starting orbd at port 9090
```

```
rebecca@ubuntu:~$ servertool -ORBInitialPort 9090 -ORBInitialHost localhost


Welcome to the Java IDL Server Tool
please enter commands at the prompt

servertool > list


        Server Id       Server Class Name              Server Application
        ---------       -----------------              ------------------

           257          sudokuserver.SudokuServer             Demo2

servertool > 
```

Code

/var/www/SudokuServer/src/sudokuserver/SudokuServer.java

```
/*
  * To change this template, choose Tools | Templates
  * and open the template in the editor.
  */
package sudokuserver;

import corbaobjects.AdminImpl;
import corbaobjects.PlayerImpl;
import java.sql.Connection;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import org.omg.CORBA.ORB;
import org.omg.CORBA.ORBPackage.InvalidName;
import org.omg.CORBA.Policy;
import org.omg.CosNaming.NameComponent;
import org.omg.CosNaming.NamingContext;
import org.omg.CosNaming.NamingContextHelper;
import org.omg.CosNaming.NamingContextPackage.AlreadyBound;
import org.omg.CosNaming.NamingContextPackage.CannotProceed;
import org.omg.CosNaming.NamingContextPackage.NotFound;
import org.omg.PortableServer.IdAssignmentPolicyValue;
import org.omg.PortableServer.LifespanPolicyValue;
import org.omg.PortableServer.POA;
import org.omg.PortableServer.POAHelper;
import org.omg.PortableServer.POAManager;
import org.omg.PortableServer.POAManagerPackage.AdapterInactive;
import org.omg.PortableServer.POAPackage.ObjectAlreadyActive;
import org.omg.PortableServer.POAPackage.ObjectNotActive;
import org.omg.PortableServer.POAPackage.ServantAlreadyActive;
import org.omg.PortableServer.POAPackage.WrongPolicy;
import org.omg.PortableServer.RequestProcessingPolicyValue;
import org.omg.PortableServer.ServantActivator;
import org.omg.PortableServer.ServantRetentionPolicyValue;

/**
 *
 * @author rebecca
 */
public class SudokuServer {

        private Connection conn = null;
        public static POA adminPOA;
        public static POA playerPOA;
        private static POA root_poa;
        private static POAManager poa_manager;
        public static ORB orb;
        public static NamingContext rootctx;
        public static EntityManagerFactory emf;


        public static void main(String[] args) {
                try {
```

```
        emf = Persistence.createEntityManagerFactory("SudokuPersistencePU");
        System.out.println("Initializing orb");
                orb = ORB.init(args, null);
                org.omg.CORBA.Object poaobj=null;
        try {
            poaobj = orb.resolve_initial_references("RootPOA");
        } catch (InvalidName ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
        }
                root_poa = POAHelper.narrow(poaobj);
                poa_manager =root_poa.the_POAManager();


                System.out.println("Publishing to name service");
                org.omg.CORBA.Object objRef=null;
        try {
            objRef = orb.resolve_initial_references("NameService");
        } catch (InvalidName ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
        }
                rootctx = NamingContextHelper.narrow(objRef);

                makeAdminPOA();
                makePlayerPOA();

                // Create a ServantActivator
                ServantActivator playerActivator =
                        new PlayerActivator();

        try {
            playerPOA.set_servant_manager(playerActivator);
        } catch (WrongPolicy ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
```

```
        }

            // Create administrator objects; and publish Corba references
            // via name service
            String adminName = "Administrator";
           // String playerTableName = "PlayerTable";

            AdminImpl anAdmin = new AdminImpl();
           // PlayerImpl aPlayer= new PlayerImpl();

            byte[] oidAdmin = adminName.getBytes();

           // byte[] oidPlayer = playerTableName.getBytes();
        try {
            adminPOA.activate_object_with_id(oidAdmin, anAdmin);
        } catch (ServantAlreadyActive ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
        } catch (ObjectAlreadyActive ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
        } catch (WrongPolicy ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
        }



            org.omg.CORBA.Object refAdmin=null;
        try {
            refAdmin = adminPOA.id_to_reference(oidAdmin);
        } catch (ObjectNotActive ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
        } catch (WrongPolicy ex) {
```

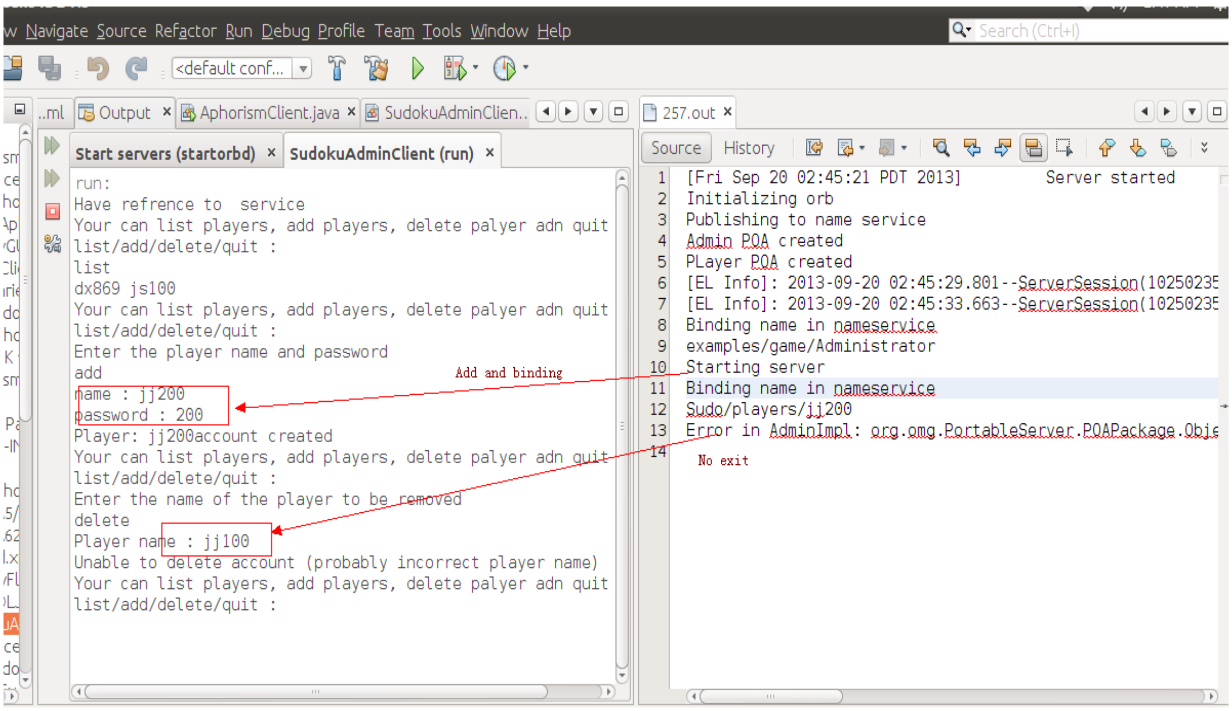
```
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
                    }
                              org.omg.CORBA.Object refPlayer=null;





                    NameComponent nc0 = new NameComponent("examples", "");
                    NameComponent nc1 = new NameComponent("game", "");
                    NameComponent nc2 = new NameComponent("Administrator", "");
                    NameComponent path[] = {nc0, nc1, nc2};
                    try {
                         registerObjWithNameService(rootctx, path, refAdmin, true);
                    } catch (InvalidName ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
                    } catch (AlreadyBound ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
                    }



                              System.out.println("Starting server");
                              poa_manager.activate();
                              orb.run();
                              System.out.println("Returned from orb.run");
                              Thread.sleep(2000);

                              // Maybe sun has closed down poa as part of orb shutdown
                          //    System.out.println("destroying orb");
                              //root_poa.destroy(true,false);
                            // orb.destroy();
                            // System.out.println("that is all gentlemen");
```

```
            } catch (CannotProceed ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
            } catch (NotFound ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
            } catch (org.omg.CosNaming.NamingContextPackage.InvalidName ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
            } catch (InterruptedException ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
            } catch (AdapterInactive ex) {
Logger.getLogger(SudokuServer.class.getName()).log(Level.SEVERE,  null, ex);
            }

        }
        private static void makePlayerPOA(){

                Policy[] playerPolicies = new Policy[4];
            playerPolicies[0] =
root_poa.create_id_assignment_policy(IdAssignmentPolicyValue.USER_ID);
            playerPolicies[1] =
root_poa.create_lifespan_policy(LifespanPolicyValue.PERSISTENT);
            playerPolicies[2] =
root_poa.create_request_processing_policy(RequestProcessingPolicyValue.USE_SERVANT_MANAGER);
            playerPolicies[3] =
root_poa.create_servant_retention_policy(ServantRetentionPolicyValue.RETAIN);
            String playerPoasName = "PlayerPOA";
            try {
```

```java
				playerPOA =
					root_poa.create_POA(playerPoasName,
					poa_manager,
					playerPolicies);

				System.out.println("PLayer POA created");
			} catch (Exception e) {
				System.out.println(e.toString());
				System.exit(1);
			}

		}
		private static void makeAdminPOA(){
			Policy[] adminPolicies = new Policy[2];

			adminPolicies[0] =
root_poa.create_id_assignment_policy(IdAssignmentPolicyValue.USER_ID);
			adminPolicies[1] =
root_poa.create_lifespan_policy(LifespanPolicyValue.PERSISTENT);
			String adminPoasName = "AdminPOA";
			try {
				adminPOA =
					root_poa.create_POA(adminPoasName,
					poa_manager,
					adminPolicies);
			} catch (Exception e) {
				System.out.println(e.toString());
				System.exit(1);
			}
			System.out.println("Admin POA created");
		}
		public static void registerObjWithNameService(NamingContext root,
				NameComponent[] serverName, org.omg.CORBA.Object obj, boolean bind) throws
				InvalidName, AlreadyBound, CannotProceed, NotFound, org.omg.CosNaming.NamingContextPackage.InvalidName {
			if (bind) {
				System.out.println("Binding name in nameservice");
			} else {
```

```
			System.out.println("Unbinding name from nameservice");
		}
		NamingContext currentContext = root;

		NameComponent[] singleElement = new NameComponent[1];

		for (int i = 0; i < serverName.length - 1; i++) {
			System.out.print(serverName[i].id + "/");
			singleElement[0] = serverName[i];
			try {
				currentContext = NamingContextHelper.narrow(
						currentContext.resolve(singleElement));

			} catch (NotFound nf) {

				currentContext =
						currentContext.bind_new_context(singleElement);
			}

		}

		singleElement[0] = serverName[serverName.length - 1];
		System.out.println(singleElement[0].id);
		if (bind) {
			currentContext.rebind(singleElement, obj);
		} else {
			currentContext.unbind(singleElement);
		}

	}
}

```

2. Admin



```
run:
Have refrence to  service
Your can list players, add players, delete palyer adn quit
list/add/delete/quit :
list
dx869 js100
Your can list players, add players, delete palyer adn quit
list/add/delete/quit :
Enter the player name and password
add
name : jj200
password : 200
Player: jj200account created
Your can list players, add players, delete palyer adn quit
list/add/delete/quit :
Enter the name of the player to be removed
delete
Player name : jj100
Unable to delete account (probably incorrect player name)
Your can list players, add players, delete palyer adn quit
list/add/delete/quit :
```

```
[Fri Sep 20 02:45:21 PDT 2013]        Server started
Initializing orb
Publishing to name service
Admin POA created
PLayer POA created
[EL Info]: 2013-09-20 02:45:29.801--ServerSession(10250235
[EL Info]: 2013-09-20 02:45:33.663--ServerSession(10250235
Binding name in nameservice
examples/game/Administrator
Starting server
Binding name in nameservice
Sudo/players/jj200
Error in AdminImpl: org.omg.PortableServer.POAPackage.Obje

```

Code

In server:

/var/www/SudokuServer/src/corbaobjects/AdminImpl.java

```
/*
  * To change this template, choose Tools | Templates
  * and open the template in the editor.
  */
package corbaobjects;

import Aphorism.AdminPOA;
import Aphorism.PlayerHelper;
import SudokuPersistence.Players;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.persistence.Query;
import org.omg.CosNaming.NameComponent;
import org.omg.PortableServer.POA;
import sudokuserver.SudokuServer;

```

```
/**
 *
 * @author rebecca
 */
public class AdminImpl extends AdminPOA {

        private EntityManagerFactory emf;
        private EntityManager em;

        public AdminImpl() {
                emf =
Persistence.createEntityManagerFactory("SudokuPersistencePU");
                em = emf.createEntityManager();
        }

        @Override
        public void shutdownServer() {

                SudokuServer.orb.shutdown(false);
        }

        private boolean registerPlayerWithNameService(String name) {
                try {
                        // Create a Corba object (but not a Player servant object)
                        // and register in name service
                        POA mypoa = SudokuServer.playerPOA;

                        byte[] oidPlayer = name.getBytes();


                        org.omg.CORBA.Object refPlayer =
                                        mypoa.create_reference_with_id(oidPlayer,
PlayerHelper.id());
                        NameComponent nc0 = new NameComponent("Sudo",
"");
                        NameComponent nc1 = new
NameComponent("players", "");
                        NameComponent nc2 = new NameComponent(name,
"");
                        NameComponent path[] = {nc0, nc1, nc2};


SudokuServer.registerObjWithNameService(SudokuServer.rootctx, path,
refPlayer, true);
```

```
                        return true;
                } catch (Exception e) {
                        System.out.println("Admin got exception while registering a name ");
                        System.out.println(e);
                        return false;
                }
        }

        private void removePlayerFromNameService(String name) {
                NameComponent nc0 = new NameComponent("Sudo", "");
                NameComponent nc1 = new NameComponent("players", "");
                NameComponent nc2 = new NameComponent(name, "");
                NameComponent path[] = {nc0, nc1, nc2};
                try {
SudokuServer.registerObjWithNameService(SudokuServer.rootctx, path, null, false);
                } catch (Exception e) {
                }
        }

        @Override
        public boolean createPlayer(String name, String password) {
                try {
                        em.getTransaction().begin();
                        Players p = new Players();
                        p.setName(name);
                        p.setPassword(password);
                        p.setCurrentmove("0");
                        em.persist(p);
                        em.getTransaction().commit();
                        registerPlayerWithNameService(name);

                        return true;
                } catch (Exception e) {

                        return false;
                }

        }

        @Override
        public String [] getnames() {
            Query q = em.createNamedQuery("Players.findAll");
             List<Players> player = (List<Players>) q.getResultList();

             String [] namelist=new String[player.size()];
             int i = 0;
            for (Players p : player) {
                namelist[i]= p.getName();
                 i++;
             }

             return namelist;
        }

        @Override
        public boolean changePassword(String name, String password) {

             try {
                 em.getTransaction().begin();
                 Query q = em.createQuery("UPDATE players SET
players.password=:password WHERE players.name=:name");
                 q.setParameter("name", name);
                 q.setParameter("password", password);
                 em.getTransaction().commit();
                 em.close();
                 return true;
             } catch (Exception e) {
                 System.out.println("Error in AdminImpl: " +
e.toString());
                 return false;
             }

        }

        @Override
        public boolean deletePlayer(String name) {
             try {

                 byte[] oidPlayer = name.getBytes();
                 SudokuServer.playerPOA.deactivate_object(oidPlayer);

                // removePlayerFromNameService(name);
                 em.getTransaction().begin();
                 Query q =
em.createNamedQuery("Players.findByName");
```

```java
            q.setParameter("name", name);
            Players p = (Players) q.getSingleResult();
            em.remove(p);
            em.getTransaction().commit();
            em.close();
            return true;
        } catch (Exception e) {

            System.out.println("Error in AdminImpl: " + e.toString());
            return false;
        }

    }

}

```

In client

/var/www/SudokuAdminClinet/src/sudokuadminclient/SudokuAdminClient.java

```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package sudokuadminclient;

import Aphorism.Admin;
import Aphorism.AdminHelper;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.logging.Level;
import java.util.logging.Logger;
import org.omg.CORBA.ORB;
import org.omg.CosNaming.NameComponent;
import org.omg.CosNaming.NamingContext;
import org.omg.CosNaming.NamingContextHelper;

/**
 *
 * @author rebecca
 */
public class SudokuAdminClient {

    private static Admin myAdmin;
    private static BufferedReader input;

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        try {

            ORB orb = ORB.init(args, null);

            org.omg.CORBA.Object objRef =
orb.resolve_initial_references("NameService");
            NamingContext initctx =
NamingContextHelper.narrow(objRef);

            NameComponent nc0 = new
NameComponent("examples", "");
            NameComponent nc1 = new NameComponent("game",
"");
            NameComponent nc2 = new
NameComponent("Administrator", "");
            NameComponent path[] = {nc0, nc1, nc2};

            myAdmin = AdminHelper.narrow(initctx.resolve(path));
            System.out.println("Have refrence to    service");


            start();

        } catch (Exception e) {
            System.out.println("Failed because : " + e);
            e.printStackTrace();
        }
    }

    private static void start() {
        try {
            String str = "l";
            while (!str.equals("q")) {
                System.out.println("Your can list players, add players, delete palyer adn quit player");
                System.out.println("list/add/delete/quit :");
                input = new BufferedReader(
                            new InputStreamReader(System.in));
                str = input.readLine();
                switch (str) {
                    case "list":
                        list();
                        break;
                    case "add":
                        add();
                        break;
                    case "delete":
                        delete();
                        break;
                    case "quit":
                        quit();
                        break;
                }

            }
        } catch (IOException ex) {
Logger.getLogger(SudokuAdminClient.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    public static    void list() {
        try{
        String    [] name = myAdmin.getnames();
        for (int i = 0; i < name.length; i++) {
            System.out.print(name[i] + " ");
        }
        System.out.println("");
        }catch (Exception e){System.err.println(e);}
    }

    public static void add() throws IOException {
        String player;
        String password;
        System.out.println("Enter the player name and password");
        System.out.print("name : ");
```

```
            player = input.readLine().trim();
            System.out.print("password : " );
            password = input.readLine().trim();

            boolean result = myAdmin.createPlayer(player, password);
            if (result) {
                System.out.println("Player: " + player + "account created");
            } else {
                System.out.println("Failed to creat account");
            }

        }

        public static void delete() {
                    try {
                            System.out.println("Enter the name of the player to be removed");
                            System.out.print("Player name : ");
                            String player = input.readLine().trim();

                            boolean result = myAdmin.deletePlayer(player);
                            if(result)
                                    System.out.println("Account deleted");
                            else
                                    System.out.println("Unable to delete account" +
                                            " (probably incorrect player name)");

                    }
                    catch(Exception e) {
                            System.out.println("Failure : " + e.toString());
                            System.exit(1);
                    }
        }

        public static   void quit() {

            myAdmin.shutdownServer();
        }
```

```
}

```

3 Player

In server

/var/www/SudokuServer/src/corbaobjects/PlayerImpl.java

```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package corbaobjects;

import Aphorism.PlayerPOA;
import SudokuPersistence.Games;
import SudokuPersistence.Moves;
import SudokuPersistence.Players;
import java.util.List;
import java.util.Random;
import javax.persistence.EntityManager;
import javax.persistence.Query;
import sudokugenerator.GeneratorClass;
import sudokugenerator.SudokuPuzzle;

import sudokuserver.SudokuServer;

/**
 *
 * @author rebecca
 */
public class PlayerImpl extends PlayerPOA {

        private String myName;
        private int gameid;
        private String aPassword;
        private String game;
        private Random r;
        private GeneratorClass sgen;
	private SudokuPuzzle sudo;
	private boolean loggedIn;
	private String savingCurrent;
	private int lev;
	private String puzzle;
	private String sol;

	public PlayerImpl(String name) {
		myName = name;
		System.out.println("PlayerImpl    starting" + name);
		loggedIn = false;
		r = new Random();
	}

	public PlayerImpl() {
		loggedIn = false;
		r = new Random();
	}

	public void loadData() {
	}

	//saving data when closed.

	public void saveData() {
		System.out.println("Saving data for " + myName);
		EntityManager em2 = SudokuServer.emf.createEntityManager();
		em2.getTransaction().begin();
		Query q = em2.createNamedQuery("Players.findByName");
		q.setParameter("name", myName);

		Players p=(Players) q.getSingleResult();
		p.setCurrentmove(game);
		em2.getTransaction().commit();
		em2.close();
	}

//handle player's    login
//1.check the password if not currect , renturn false
//2.if login success, set variable game and gameid 's value;
	@Override
	public boolean login(String password) {
        EntityManager em = SudokuServer.emf.createEntityManager();
        Query q = em.createNamedQuery("Players.findByName");
        q.setParameter("name", myName);
        Players player = (Players) q.getSingleResult();
        if (password.equals(player.getPassword())) {
            if (player.getGameid() == 0) {
                game = "0";
            } else {
                gameid = player.getGameid();
                game = player.getCurrentmove();

                findPuzzle();

            }

            loggedIn = true;
            return true;
        } else {
            return false;
        }

    }
// find the origanal puzzle
// set value to private var puzzle
//purpose is to give client to set the origanla number uneditable.

    public void findPuzzle() {
        EntityManager em = SudokuServer.emf.createEntityManager();
        Query q = em.createNamedQuery("Games.findByGameid");
        q.setParameter("gameid", gameid);
        Games g = (Games) q.getSingleResult();
        puzzle = g.getPuzzle();
        sol = g.getSolution();
    }
// this method will be called only when people just login
// the purpose for this method if to return a string to show palyer's old record
// in table : Games there are record with level and players name and if it is finithed.
//if player never finish any game, then return You have not completed any games.

```

```java
    @Override
    public String getPuzzle() {
        return puzzle;
    }

    @Override
    public String gamerecord() {

        int[] record = new int[5];
        String message = "You have completed: ";
        record[0] = 0;
        record[1] = 0;
        record[2] = 0;
        record[3] = 0;
        record[4] = 0;
        EntityManager em = SudokuServer.emf.createEntityManager();
        Query q = em.createNamedQuery("Games.findByPlayer");
        q.setParameter("player", myName);
        List<Games> games = (List<Games>) q.getResultList();
        if (!game.isEmpty()) {
            for (Games g : games) {
                if (g.getFnish() == 1) {
                    record[g.getLevel()]++;
                }

            }

            if (record[0] != 0) {
                message = message + record[0] + " very easy games ";
            }
            if (record[1] != 0) {
                message = message + record[1] + "    easy games ";
            }
            if (record[2] != 0) {
                message = message + record[2] + " medium games ";

            }
            if (record[3] != 0) {
                message = message + record[3] + " hard games ";
            }
            if (record[4] != 0) {
                    message = message + record[4] + " hell games ";

                }

                if (message.equals("You have completed: ")) {
                    message = "You have not completed any game.";
                }
            } else {
                message = "You have not completed any game.";
            }
            return message;
        }
// just return game string

        @Override
        public String loadGame() {
            return game;
        }
//this will be used when player wants start a new game.
//call the method so a sudo will be settle.
// at the end of this method will call saveNewGame()
// just make the code more tidy, saveNewGame will handle the database
// some part will be changed for database.

        @Override
        public String create(short level) {
            lev = level;
            sgen = new GeneratorClass();
            sudo = null;
            switch (level) {
                case 0:
                    sudo = sgen.generateVeryEasy();
                    break;
                case 1:
                    sudo = sgen.generateEasy();
                    break;
                case 2:
                    sudo = sgen.generateAverage();
                    break;
                case 3:
                    sudo = sgen.generateDifficult();
                    break;
                case 4:
                    sudo = sgen.generateDevilishlyHard();
                    break;
            }

            String a = sudo.getPuzlString();

            saveNewGame();
            return a;
        }
        //will be called by create method
        //take none para becuase all the var is updated.
        //just put information into database
        //1. int games, new row will be recoard.
        //2. in players , player's gameid   & currentmove will be updated

        public void saveNewGame() {
            EntityManager em = SudokuServer.emf.createEntityManager();
            puzzle = sudo.getPuzlString();
            sol = sudo.getSolnString();
            em.getTransaction().begin();
            Games g = new Games();
            g.setFnish(0);
            g.setPlayer(myName);
            g.setLevel(lev);
            g.setPuzzle(sudo.getPuzlString());
            g.setSolution(sudo.getSolnString());
            em.persist(g);
            em.getTransaction().commit();
            em.close();
            EntityManager em2 = SudokuServer.emf.createEntityManager();
            em2.getTransaction().begin();
            Query q = em2.createQuery("SELECT max(g.gameid) FROM Games g");
            gameid = (int) q.getSingleResult();
            Query q2 = em2.createNamedQuery("Players.findByName");
            q2.setParameter("name", myName);
            Players player = (Players) q2.getSingleResult();

            player.setGameid(gameid);
            player.setCurrentmove(sudo.getPuzlString());
            em2.getTransaction().commit();
            em2.close();

```

```java
    }

    @Override
    public boolean recordMove(String string) {
        char a[] = string.toCharArray();
        int row = Integer.parseInt(a[0] + "");
        int col = Integer.parseInt(a[1] + "");

        StringBuffer str = new StringBuffer(game);
        str.setCharAt(row * 9 + col, a[2]);
        game = str.toString();

        EntityManager em = SudokuServer.emf.createEntityManager();
        em.getTransaction().begin();
        Moves move = new Moves();
        move.setGameid(gameid);
        move.setPlayer(myName);
        move.setMove(string);

        em.persist(move);
        em.getTransaction().commit();
        em.close();

        System.out.println("New move has been record: " + string);

        EntityManager em2 = SudokuServer.emf.createEntityManager();
        em2.getTransaction().begin();
        Query q = em2.createNamedQuery("Players.findByName");
        q.setParameter("name", myName);

        Players p=(Players) q.getSingleResult();
        p.setCurrentmove(game);
        em2.getTransaction().commit();
        em2.close();
        return true;
    }
//check the data
    //if the updated string is equal to solution then return true
    //at the same time remove all the moves for this game

    @Override
    public boolean check(String string) {
        System.out.println("Method check called by client");
        if (string.equals(sol)) {
            System.out.println("same as solution");
            EntityManager em = SudokuServer.emf.createEntityManager();
            em.getTransaction().begin();
            Query q = em.createNamedQuery("Moves.findByPlayer");
            q.setParameter("player", myName);
            List<Moves> moves = (List<Moves>) q.getResultList();
            if (!moves.isEmpty()) {
                for (Moves m : moves) {
                    em.remove(m);
                }
                em.getTransaction().commit();
                em.close();
            }
            return true;
        }
        return false;
    }
//find the biggest timestamp for this game, and delete row
    //before delete give client the string of move

    @Override
    public String undo() {
        String str = "";
        EntityManager em = SudokuServer.emf.createEntityManager();
        em.getTransaction().begin();
        Query q = em.createQuery("SELECT    m from Moves m where m.gameid=:gameid and m.player=:player ORDER BY m.time desc");
        q.setParameter("gameid", gameid);
        q.setParameter("player", myName);
        List<Moves> moves = (List<Moves>) q.getResultList();

        str = moves.get(0).getMove();
        em.remove(moves.get(0));
        em.getTransaction().commit();
        em.close();
        System.out.println("Move: " + str + "has been undoed.");
        return str;
    }
}
```

Player Activator

/var/www/SudokuServer/src/sudokuserver/PlayerActivator.java

```java
/*
  * To change this template, choose Tools | Templates
  * and open the template in the editor.
  */
package sudokuserver;

import corbaobjects.PlayerImpl;
import org.omg.CORBA.LocalObject;
import org.omg.PortableServer.ForwardRequest;
import org.omg.PortableServer.POA;
import org.omg.PortableServer.Servant;
import org.omg.PortableServer.ServantActivator;

/**
  *
  * @author rebecca
  */
public class PlayerActivator extends LocalObject implements
ServantActivator {

        @Override
        public Servant incarnate(byte[] oid, POA adapter) throws
ForwardRequest {
                String name = new String(oid);
                System.out.println("Activating Player " + name);
              PlayerImpl gpi = new PlayerImpl(name);

                return gpi;
        }

        @Override
        public void etherealize(byte[] oid,
                        POA adapter,
                        Servant serv,
                    boolean cleanup_in_progress,
                    boolean remaining_activations) {
                String name = new String(oid);
                System.out.println("Etherealizing Player " + name);
               PlayerImpl gpi = (PlayerImpl) serv;
               gpi.saveData();
                System.out.println(name + " should have saved data");
        }
}

```

In client

/var/www/AphorismClient/src/aphorismclient/AphorismClient.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package aphorismclient;

import Aphorism.Player;
import Aphorism.PlayerHelper;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import myGUI.ClientGUI;
import org.omg.CORBA.ORB;
import org.omg.CosNaming.NameComponent;
import org.omg.CosNaming.NamingContext;
import org.omg.CosNaming.NamingContextHelper;

/**
 *
 * @author Administrator
 */
public class AphorismClient {
        //private static final String filename="F:\\web\\398\\AphorismServer\\src\\Aphorism.ref";
        private static Player myPlayer;
        private static String name;
        /**
         * @param args the command line arguments
         */
        public static void main(String[] args) {
```

```java
        try {

            BufferedReader input = new BufferedReader(
                    new InputStreamReader(System.in));
            System.out.print("Enter player name : ");
            name = input.readLine().trim();
            ORB orb = ORB.init(args, null);
            org.omg.CORBA.Object objRef = orb.resolve_initial_references("NameService");
            NamingContext initctx = NamingContextHelper.narrow(objRef);
            NameComponent nc0 = new NameComponent("Sudo", "");
            NameComponent nc1 = new NameComponent("players", "");
            NameComponent nc2 = new NameComponent(name, "");
            NameComponent path[] = {nc0, nc1, nc2};

            myPlayer = PlayerHelper.narrow(initctx.resolve(path));
            System.out.println("Have refrence to    service");


            /**
             * *****for assignment 2 *********
             */
            launchGUI();
            /**
             * *****end of assignment 2 *********
             */
        } catch (Exception e) {
            System.out.println("Failed because : " + e);
        }
    }
    //start GUI

    private static void launchGUI() {
        java.awt.EventQueue.invokeLater(new Runnable() {
            @Override
            public void run(){
                //show the GUI
                new ClientGUI(name,myPlayer).setVisible(true);
            }
        });
    }
    public Player getPlayer(){
    return myPlayer;
    }

}

```

For my GUI

Code is

/var/www/AphorismClient/src/myGUI/ClientGUI.java

```java
package myGUI;

import Aphorism.Player;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import javax.swing.event.TableModelEvent;
import javax.swing.event.TableModelListener;
import javax.swing.table.DefaultTableModel;

/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author rebecca
 */
public class ClientGUI extends javax.swing.JFrame {

    private static String name;
    private Player myPlayer;
    private String game;
    private String puzzle = "";
    int solved = 0;

    /**
     * Creates new form Client
     */
    public ClientGUI(String tempname, Player player) {
            this.addWindowListener(new WindowAdapter(){
   @Override
   public void windowClosing(WindowEvent we){

    System.exit(0);
   }
  });
        name = tempname;
        myPlayer = player;
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // handle login
        //pass password value to check
        //log a game there is a unfinithsed game
        //other wise just waiting for the pplayer choose one.
        String logName = login.getText();
        String password = pwd.getText().trim();

        if (logName.equals(name) && myPlayer.login(password)) {
            System.out.println("login seccuss.");
            game = myPlayer.loadGame();
            System.out.println("loading game:" + game);
            String str = myPlayer.gamerecord();
            if (!"0".equals(game)) {
                System.out.println("there is an unfinished game here");
                System.out.println("going to settle table.");
                puzzle = myPlayer.getPuzzle();
                setTable();

            }
```

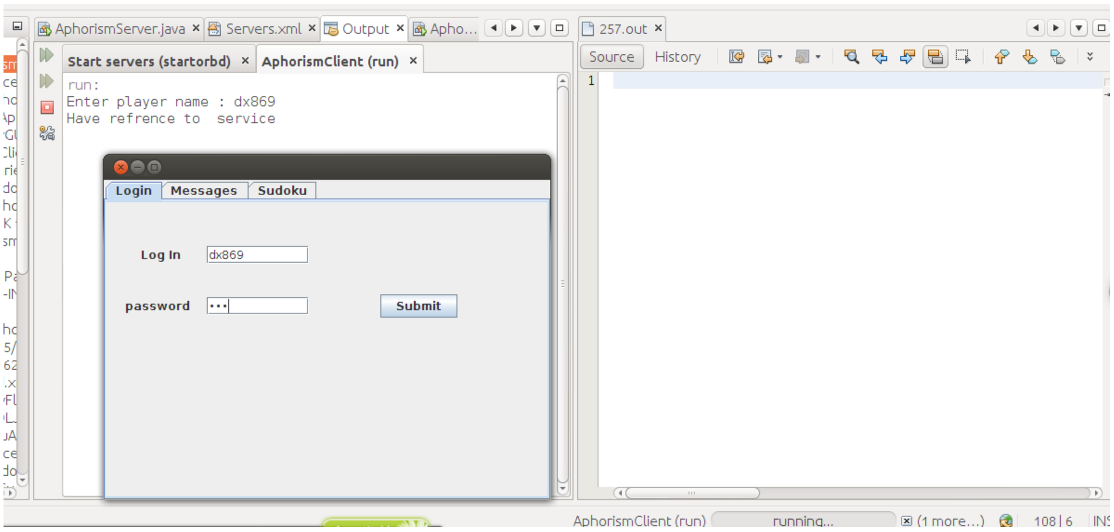
```


                message.setText(str);
                jTabbedPane1.setSelectedComponent(jPanel2);


            } else {
                System.out.println("Wrong password");
                System.exit(0);
            }
        }
        //set table
        //1.set the table as current move
        //2.set the origanla data to unedited.

        private void setTable() {
            solved = 0;
            //here i just override the isCellEditable so if the number is
origanla puzzle number
            //player can not change it.
            DefaultTableModel mt = new DefaultTableModel() {
                @Override
                public boolean isCellEditable(int row, int column) {
                    if (!"0".equals(game)) {
                        char puz[] = puzzle.toCharArray();
                        if (puz[row * 9 + column] != '0') {
                            return false;
                        }
                        return true;
                    }

                    return true;
                }
            };
            mt.setColumnCount(9);
            mt.setRowCount(9);

//set uneditable.
            if (!"0".equals(game)) {

                char cha[] = game.toCharArray();
                for (int i = 0; i < 81; i++) {
                    if (cha[i] != '0') {
                        mt.setValueAt(cha[i], i / 9, i % 9);
                        solved++;

                    }

                }

                table.setModel(mt);
                System.out.println("Table settled");


            }


//a listener that if data changed, the value will be read and record;
            table.getModel().addTableModelListener(new TableModelListener() {
                @Override
                public void tableChanged(TableModelEvent e) {
                    if (e.getType() == TableModelEvent.UPDATE) {

                        int row = e.getLastRow();
                        int col = e.getColumn();

                        String newvalue = table.getValueAt(e.getLastRow(), e.getColumn()).toString();
                       //update game varibal.
                        char a[]=newvalue.toCharArray();
                        StringBuffer str1 = new StringBuffer(game);
                        str1.setCharAt(row * 9 + col, a[0]);
                        game = str1.toString();
                        String str = Integer.toString(row) + Integer.toString(col) + newvalue;

                        System.out.println("calling recordmove method to record this move: "+ newvalue);
                        myPlayer.recordMove(str);
                        solved++;
                        // if player finished the game then print out.
                        if (solved > 80) {
                            if (myPlayer.check(game)) {

                                message.setText("Congradulations ! You just finish a game");
```

```
                            jTabbedPane1.setSelectedComponent(jPanel2);

                        }
                    }

                }

            }
        });

    }
    private void pwdActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        //select level to get a new game
        short a = 5;
        if (veryEasy.isSelected()) {
            a = 0;
        } else if (easy.isSelected()) {
            a = 1;
        } else if (medium.isSelected()) {
            a = 2;
        } else if (hard.isSelected()) {
            a = 3;
        } else if (hell.isSelected()) {
            a = 4;
        }
        game = myPlayer.create(a);
        puzzle = myPlayer.getPuzzle();
        setTable();


    }

    private void tableInputMethodTextChanged(java.awt.event.InputMethodEvent evt) {
        // TODO add your handling code here:
    }
```

```java

    private void tableKeyReleased(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
    }

    private void undoActionPerformed(java.awt.event.ActionEvent evt) {
//undo method
        //server return move should be delete
        //change the string then setTable again

        String undoStr = myPlayer.undo();
        char a[] = undoStr.toCharArray();
        int row = Integer.parseInt(a[0] + "");
        int col = Integer.parseInt(a[1] + "");
        StringBuffer str = new StringBuffer(game);
        str.setCharAt(row * 9 + col, '0');
        game = str.toString();
        setTable();
        solved--;

    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(ClientGUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(ClientGUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(ClientGUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(ClientGUI.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
//        java.awt.EventQueue.invokeLater(new Runnable() {
//            public void run() {
//                new ClientGUI(name).setVisible(true);
//            }
//        });
    }
    // Variables declaration - do not modify
    // End of variables declaration
}

```

Display

Before login

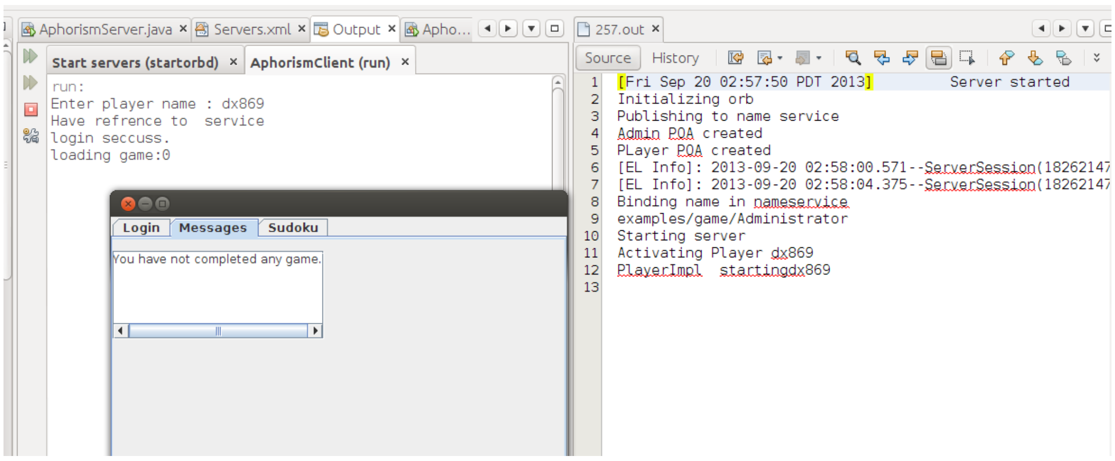


```
run:
Enter player name : dx869
Have refrence to  service
```

Log In: dx869

password: •••

After login this example there is no games record for the person

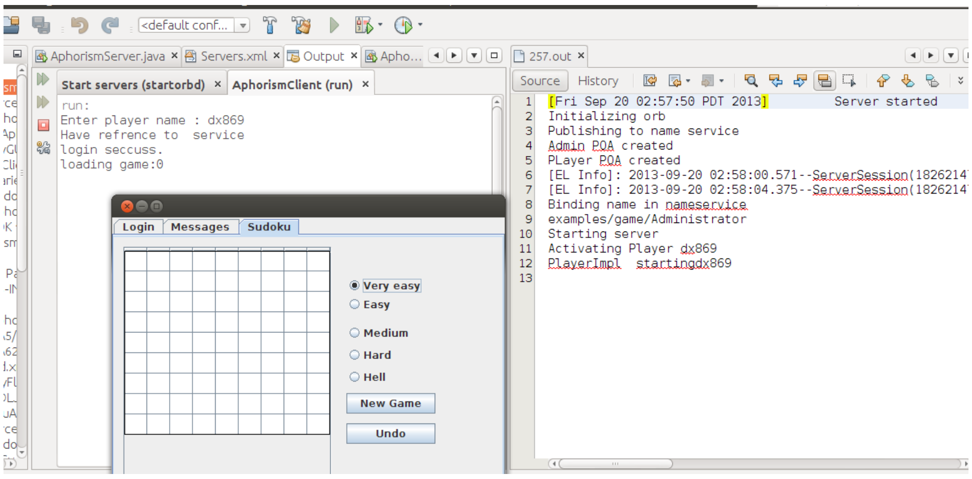


```
run:
Enter player name : dx869
Have refrence to  service
login seccuss.
loading game:0
```

You have not completed any game.

```
[Fri Sep 20 02:57:50 PDT 2013]         Server started
Initializing orb
Publishing to name service
Admin POA created
PLayer POA created
[EL Info]: 2013-09-20 02:58:00.571--ServerSession(18262147
[EL Info]: 2013-09-20 02:58:04.375--ServerSession(18262147
Binding name in nameservice
examples/game/Administrator
Starting server
Activating Player dx869
PlayerImpl  startingdx869
```

That is why the Sudoku part will look like this.



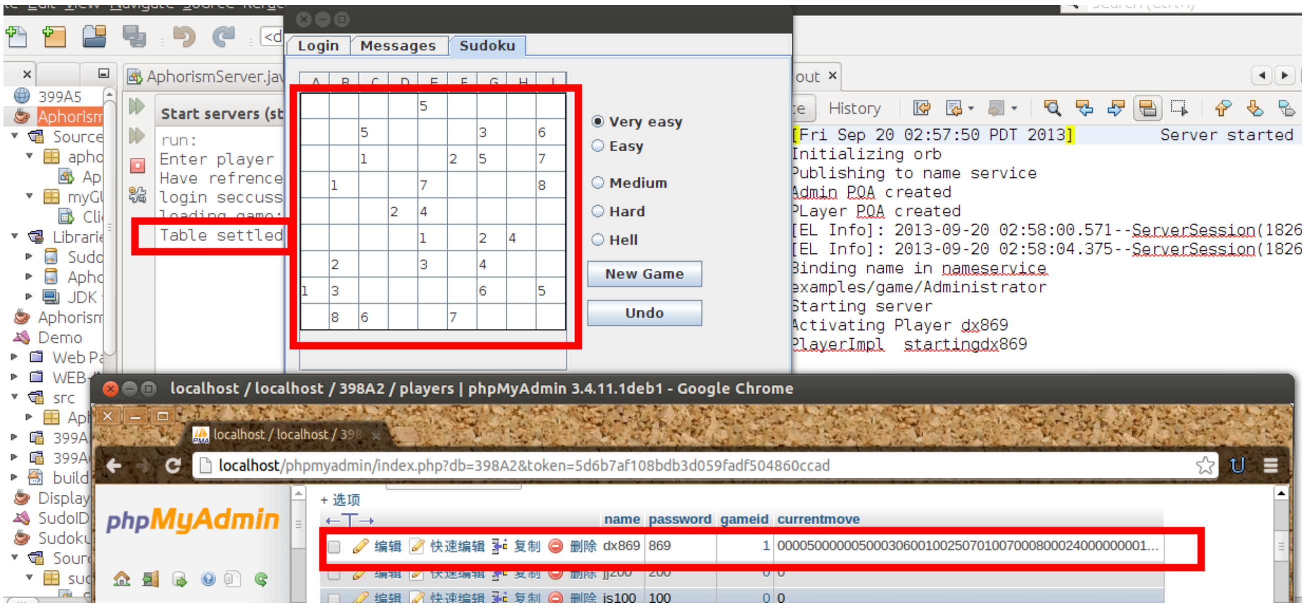
```
run:
Enter player name : dx869
Have refrence to  service
login seccuss.
loading game:0
```

[✓ Very easy] Easy Medium Hard Hell

```
[Fri Sep 20 02:57:50 PDT 2013]         Server started
Initializing orb
Publishing to name service
Admin POA created
PLayer POA created
[EL Info]: 2013-09-20 02:58:00.571--ServerSession(1826214
[EL Info]: 2013-09-20 02:58:04.375--ServerSession(1826214
Binding name in nameservice
examples/game/Administrator
Starting server
Activating Player dx869
PlayerImpl  startingdx869
```

But if you chose a level of game

Table will be settled and database will be updated.



```
run:
Enter player
Have refrence
login seccuss
loading game:
Table settled
```

A	B	C	D	E	F	G	H	I
				5				
		5				3		6
		1			2	5		7
	1			7				8
			2	4				
				1		2	4	
	2			3		4		
1	3					6		5
	8	6			7			

```
[Fri Sep 20 02:57:50 PDT 2013]            Server started
Initializing orb
Publishing to name service
Admin POA created
PLayer POA created
[EL Info]: 2013-09-20 02:58:00.571--ServerSession(1826
[EL Info]: 2013-09-20 02:58:04.375--ServerSession(1826
Binding name in nameservice
examples/game/Administrator
Starting server
Activating Player dx869
PlayerImpl  startingdx869
```

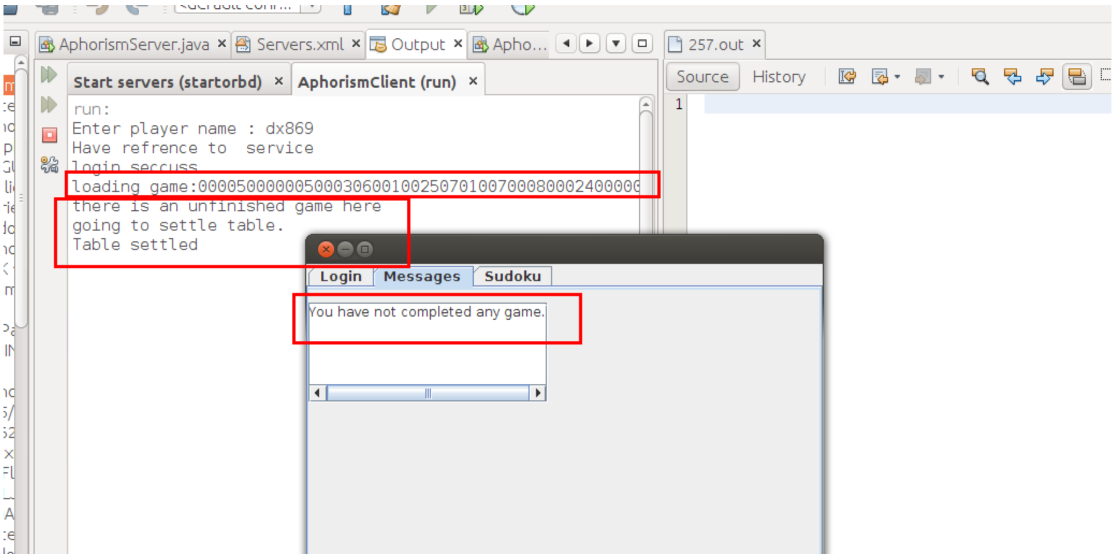
localhost / localhost / 398A2 / players | phpMyAdmin 3.4.11.1deb1 - Google Chrome

localhost/phpmyadmin/index.php?db=398A2&token=5d6b7af108bdb3d059fadf504860ccad

name	password	gameid	currentmove
dx869	869	1	00005000000050003060010025070100700080002400000001...
jj200	200	0	0
is100	100	0	0

If you quit it and restart it

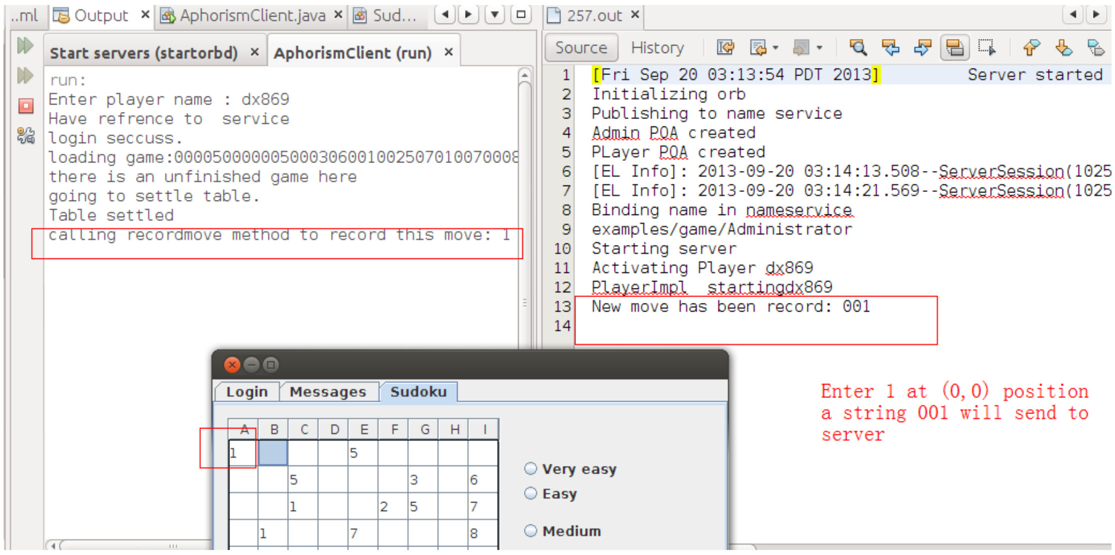
Data will be load after login



```
run:
Enter player name : dx869
Have refrence to  service
login seccuss
loading game:0000500000005000306001002507010070008000240000
there is an unfinished game here
going to settle table.
Table settled
```

You have not completed any game.

Enter a value will looks like that: as you can see the data updated in table, client record the value and sent to server.



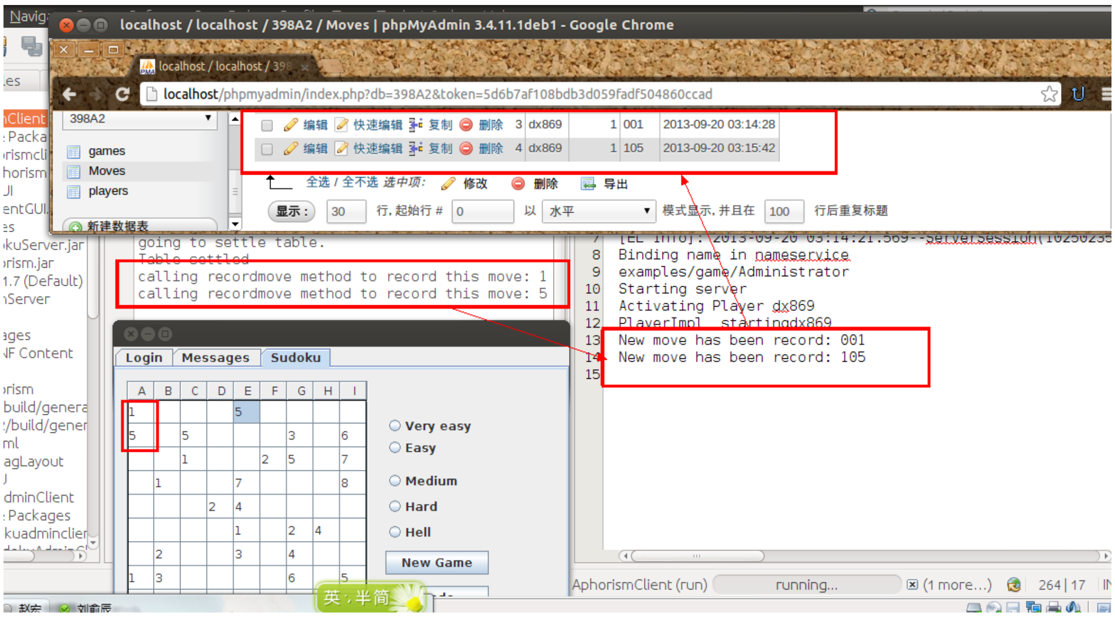
```
run:
Enter player name : dx869
Have refrence to  service
login seccuss.
loading game:0000500000050003060010025070100700008
there is an unfinished game here
going to settle table.
Table settled
calling recordmove method to record this move: 1
```

```
[Fri Sep 20 03:13:54 PDT 2013]          Server started
Initializing orb
Publishing to name service
Admin POA created
PLayer POA created
[EL Info]: 2013-09-20 03:14:13.508--ServerSession(1025
[EL Info]: 2013-09-20 03:14:21.569--ServerSession(1025
Binding name in nameservice
examples/game/Administrator
Starting server
Activating Player dx869
PlayerImpl  startingdx869
New move has been record: 001
```

A	B	C	D	E	F	G	H	I
1				5				
		5				3		6
		1			2	5		7
	1			7				8

Enter 1 at (0,0) position
a string 001 will send to
server

Enter 5 at (1,0) position, the database will update according to the timestamp.



localhost / localhost / 398A2 / Moves | phpMyAdmin 3.4.11.1deb1 - Google Chrome

localhost/phpmyadmin/index.php?db=398A2&token=5d6b7af108bdb3d059fadf504860ccad

3	dx869	1	001	2013-09-20 03:14:28
4	dx869	1	105	2013-09-20 03:15:42

```
going to settle table.
Table settled
calling recordmove method to record this move: 1
calling recordmove method to record this move: 5
```

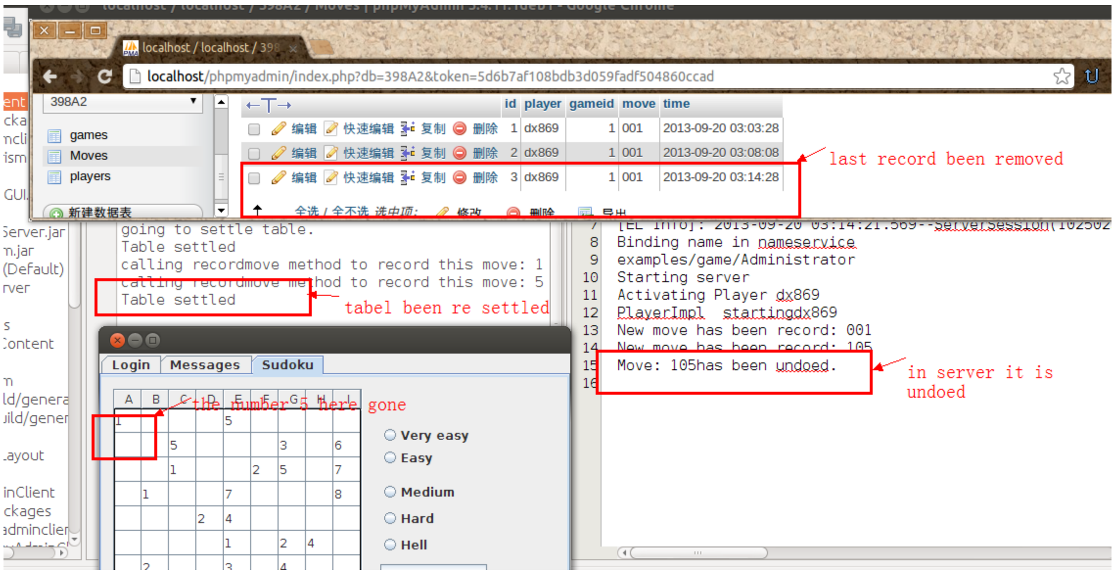
```
Binding name in nameservice
examples/game/Administrator
Starting server
Activating Player dx869
PlayerImpl  startingdx869
New move has been record: 001
New move has been record: 105
```

A	B	C	D	E	F	G	H	I
1				5				
5		5				3		6
		1			2	5		7
	1			7				8
			2	4				
				1		2	4	
	2			3		4		
1	3					6		5

Undo

Last data which is enter 5 at (1,0) has been deleted from database.

In the table , 5 is gone too.



localhost/phpmyadmin/index.php?db=398A2&token=5d6b7af108bdb3d059fadf504860ccad

id	player	gameid	move	time
1	dx869	1	001	2013-09-20 03:03:28
2	dx869	1	001	2013-09-20 03:08:08
3	dx869	1	001	2013-09-20 03:14:28

```
going to settle table.
Table settled
calling recordmove method to record this move: 1
calling recordmove method to record this move: 5
Table settled
```

```
[EL Info]: 2013-09-20 03:14:21.569--ServerSession(102502
Binding name in nameservice
examples/game/Administrator
Starting server
Activating Player dx869
PlayerImpl  startingdx869
New move has been record: 001
New move has been record: 105
Move: 105has been undoed.
```

Aphorism.idl

```
module Aphorism {

typedef sequence<string> stringlist;

      interface Admin {



            boolean createPlayer(in string name, in string password);

            boolean changePassword(in string name, in string password);

            boolean deletePlayer(in string name);

            void shutdownServer();

                              stringlist getnames();



      };

          interface Player{

                              string create(in short level);
```

```
                    string gamerecord();
                    string loadGame();
                    boolean login(in string password);
                    boolean recordMove(in string move);
                    boolean check(in string answer);
                    string undo();
                    string getPuzzle();
};
};
```

The Server.xml is same as exercise 2.