import Inter.\*;

import java.util.Vector;

import java.util.Iterator;

public class DBControlServerImpl extends DBControlServerPOA

{

private Vector clients = new Vector(); // The clients currently connected

private CallingThread ct = null;

public DBControlServerImpl()

{

ct = new CallingThread(this);

}

public void requestConnect(DBClientListener dbcl)

{

System.out.println("A client has connected.");

clients.add(dbcl);

}

public void startCallingThread()

{

ct.start();

}

public int getClientCount()

{

return clients.size();

}

public void sendWorkMessages()

{

Iterator it = clients.iterator();

while (it.hasNext())

{

DBClientListener dbcl = (DBClientListener) it.next();

dbcl.doWork();

}

}

}

import Inter.\*;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

public class DBClientListenerImpl extends DBClientListenerPOA

{

public void doWork()

{

File rDir = new File(System.getProperty("user.dir"));

rDir = new File(rDir.getAbsolutePath() + File.separatorChar);

File inFile = new File(rDir.getAbsolutePath() + File.separatorChar

+ "Database.dat");

if (inFile.exists())

{

try

{

BufferedReader in = new BufferedReader(new FileReader(inFile));

String tmp = ""; // Temp line holder.

long timeElapsed = 0;

long startTime = System.currentTimeMillis();

tmp = in.readLine();

while (tmp != null)

{

tmp = in.readLine();

}

long endTime = System.currentTimeMillis();

timeElapsed = endTime - startTime;

System.out.println("Client finished computations on file in "

+ timeElapsed + " ms.");

} catch (Exception e)

{

e.printStackTrace();

}

} else {

System.out.println("Database file was not found");

}

}

}

import org.omg.CORBA.ORB;

import org.omg.PortableServer.POA;

import org.omg.PortableServer.POAHelper;

import Inter.\*;

public class Client

{

public static void main(String[] args)

{

try

{

// initialize orb

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

System.out.println("Initialized ORB");

// Instantiate Servant and create reference

POA rootPOA = POAHelper.narrow(orb

.resolve\_initial\_references("RootPOA"));

DBClientListenerImpl listener = new DBClientListenerImpl();

rootPOA.activate\_object(listener);

DBClientListener ref = DBClientListenerHelper.narrow(rootPOA

.servant\_to\_reference(listener));

// Resolve DBControlServer

DBControlServer dbcServer = DBControlServerHelper

.narrow(orb.string\_to\_object(

"corbaname:iiop:1.2@localhost:1050#DBControlServer"));

// Register listener reference (callback object) with MessageServer

dbcServer.requestConnect(ref);

System.out.println("Registered with DBControlServer.");

// Activate rootPOA

rootPOA.the\_POAManager().activate();

// Wait for work request

System.out.println("Waiting for work request.");

orb.run();

} catch (Exception e)

{

e.printStackTrace();

}

}

}

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class CallingThread extends Thread

{

DBControlServerImpl dbcsiImpl = null;

public CallingThread(DBControlServerImpl dbcsiImpl)

{

this.dbcsiImpl = dbcsiImpl;

}

public void run()

{

int userInput = 0;

try

{

BufferedReader read = new BufferedReader(new InputStreamReader(

System.in));

try

{

System.out.print("Please enter the number of clients you wish to connect: ");

userInput = Integer.parseInt(read.readLine());

System.out.println("Waiting for " + userInput + " clients to connect...");

} catch (Exception e)

{

e.printStackTrace();

}

boolean completed = false;

for (;;)

{

if (dbcsiImpl.getClientCount() >= userInput && !completed)

{

System.out.println(userInput + " clients connected, starting work; check client screens...");

dbcsiImpl.sendWorkMessages();

completed = true;

}

}

} catch (Exception e)

{

e.printStackTrace();

}

}

}

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class CallingThread extends Thread

{

DBControlServerImpl dbcsiImpl = null;

public CallingThread(DBControlServerImpl dbcsiImpl)

{

this.dbcsiImpl = dbcsiImpl;

}

public void run()

{

int userInput = 0;

try

{

BufferedReader read = new BufferedReader(new InputStreamReader(

System.in));

try

{

System.out.print("Please enter the number of clients you wish to connect: ");

userInput = Integer.parseInt(read.readLine());

System.out.println("Waiting for " + userInput + " clients to connect...");

} catch (Exception e)

{

e.printStackTrace();

}

boolean completed = false;

for (;;)

{

if (dbcsiImpl.getClientCount() >= userInput && !completed)

{

System.out.println(userInput + " clients connected, starting work; check client screens...");

dbcsiImpl.sendWorkMessages();

completed = true;

}

}

} catch (Exception e)

{

e.printStackTrace();

}

}

}