**Clients - Binder – Servers**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package appservices;

import java.sql.\*;

/\*\*

\*

\* @author Admin

\*/

public class AddServer {

/\*\*

\* @param args the command line arguments

\*/

private static boolean addServer(String query) throws Exception {

Class.forName("oracle.jdbc.driver.OracleDriver");

Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE", "system", "hr");

try {

con.createStatement().executeUpdate(query);

return true;

} catch (Exception e) {

return false;

}

}

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

String serverIP = "'localhost'";

String serverName = "'TranslatorServer'";

String serverPort = "'8778'";

//String createTable = "create table servers (servername varchar2(20) , serverport varchar2(20) , serverip varchar2(20))";

String query = "insert into servers values(" + serverName +','+ serverPort +','+ serverIP + ")";

if (addServer(query))

System.out.println("Server Added");

else

System.out.println("Server add failed");

}

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package appservices;

import java.io.\*;

import java.net.Socket;

import java.util.Scanner;

import java.util.concurrent.\*;

/\*\*

\*

\* @author Admin

\*/

public class AppServices { // client class

/\*\*

\* @param args the command line arguments

\*/

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

// TODO code application logic here

Scanner input = new Scanner(System.in);

System.out.println("Enter server name : ");

//String serverName = "chatServer";

String serverName = input.nextLine();

ExecutorService service = Executors.newFixedThreadPool(1);

Binder binder = new Binder(serverName);

Future task = service.submit(binder);

String serverPort = (String) task.get();

if (serverPort == null) {

serverPort = "";

}

while (true) {

switch (serverPort) {

case "8778": { //TranslatorServer

System.out.println("PORT : " + serverPort);

try {

Socket soc = new Socket("localhost", 8778);

BufferedWriter bufw = new BufferedWriter(new OutputStreamWriter(soc.getOutputStream()));

BufferedReader bufr = new BufferedReader(new InputStreamReader(soc.getInputStream()));

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

String rec = bufr.readLine();

if (rec != null) {

System.out.println(rec);

}

String data = bufrc.readLine();

if (data != null) {

bufw.write(data);

bufw.newLine();

bufw.flush();

}

rec = bufr.readLine();

if (rec != null) {

System.out.println(rec);

}

//data = bufrc.readLine();

//bufw.write(data);

//bufw.newLine();

bufw.flush();

soc.close();

} catch (Exception e) {

System.out.println("The server is down!");

break;

}

break;

}

case "5472": { //chatServer

System.out.println("PORT : " + serverPort);

try {

Socket soc = new Socket("localhost", 5472);

BufferedWriter bufw = new BufferedWriter(new OutputStreamWriter(soc.getOutputStream()));

BufferedReader bufr = new BufferedReader(new InputStreamReader(soc.getInputStream()));

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

String data = bufrc.readLine();

while (!data.equals("bye")) {

if (data != null) {

bufw.write(data);

bufw.newLine();

bufw.flush();

}//if

String rec = bufr.readLine();

if (rec != null) {

System.out.println(rec);

}

data = bufrc.readLine();

}

bufw.write(data);

bufw.newLine();

bufw.flush();

soc.close();

} catch (Exception e) {

System.out.println("The server is down!");

break;

}

break;

}

case "5555": { //postsServer

System.out.println("PORT : " + serverPort);

Socket soc = new Socket("localhost", 5555);

break;

}

default: {

System.out.println("There is no server with this name");

}

}

System.out.println("Enter server name : ");

serverName = input.nextLine();

service = Executors.newFixedThreadPool(1);

binder = new Binder(serverName);

task = service.submit(binder);

serverPort = (String) task.get();

if (serverPort == null) {

serverPort = "";

}

}

}

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package appservices;

import java.sql.\*;

import java.util.concurrent.\*;

/\*\*

\*

\* @author Admin

\*/

public class Binder implements Callable<String> {

public Binder() throws SQLException {

}

private String serverName;

public Binder(String serverName) {

this.serverName = serverName;

}

private static ResultSet dbResult(String query) throws Exception {

Class.forName("oracle.jdbc.driver.OracleDriver");

Connection con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE", "system", "hr");

return con.createStatement().executeQuery(query);

}

private static boolean checkIfExist(String serverName) throws Exception {

ResultSet rs = dbResult("select \* from servers where servername ='" + serverName + "'");

if (rs.next()) {

return true;

}

return false;

}

@Override

public String call() throws Exception {

if (checkIfExist(serverName)) {

ResultSet rs = dbResult("select \* from servers where servername ='" + serverName + "'");

rs.next();

return rs.getString("SERVERPORT");

}

return null;

}

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package appservices;

import java.io.\*;

import java.net.\*;

import java.util.\*;

class Translator extends Thread {

Socket con;

public Translator(Socket con) {

this.con = con;

}

private static String getPageSource(String textToTranslate, String translateFrom, String translateTo) throws Exception {

String pageUrl = String.format("https://translate.google.com/m?sl=%s&tl=%s&q=%s", translateFrom, translateTo, URLEncoder.encode(textToTranslate, "UTF-8"));

//System.out.println(pageUrl);

URL url = new URL(pageUrl);

HttpURLConnection connection = null;

BufferedReader bufferedReader = null;

StringBuilder pageSource = new StringBuilder();

try {

connection = (HttpURLConnection) url.openConnection();

connection.setConnectTimeout(15000);

connection.setRequestProperty("User-Agent", "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.11 (KHTML, like Gecko) Chrome/23.0.1271.95 Safari/537.11");

bufferedReader = new BufferedReader(new InputStreamReader(connection.getInputStream(), "UTF-8"));

String line;

while ((line = bufferedReader.readLine()) != null) {

pageSource.append(line + System.lineSeparator());

}

return pageSource.toString();

} catch (Exception e) {

e.printStackTrace();

} finally {

if (connection != null) {

connection.disconnect();

}

if (bufferedReader != null) {

bufferedReader.close();

}

}

return null;

}

public static String translate(String textToTranslate, String translateFrom, String translateTo) throws Exception {

try {

String pageSource = getPageSource(textToTranslate, translateFrom, translateTo);

String[] translatedText1 = pageSource.split("\"result-container\">");

String[] translatedText2 = translatedText1[1].split("</");

return translatedText2[0];

} catch (Exception e) {

return "There was an error translating the text.";

}

}

public void run() {

try {

System.out.println("Client " + TranslatorServer.clientNum + " is connected now!");

BufferedReader bufr = new BufferedReader(new InputStreamReader(con.getInputStream()));

BufferedWriter bufw = new BufferedWriter(new OutputStreamWriter(con.getOutputStream()));

String textToTranslate;

bufw.write("Enter text: ");

bufw.newLine();

bufw.flush();

textToTranslate = bufr.readLine();

String translatedText = Translator.translate(textToTranslate, Language.ARABIC, Language.ENGLISH);

bufw.write("Translated text: " + translatedText);

bufw.newLine();

bufw.flush();

con.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

public class TranslatorServer {

static int clientNum = 0;

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

try {

ServerSocket translator = new ServerSocket(8778);

System.out.println("Translator Server is running now ...");

while (true) {

new Translator(translator.accept()).start();

}

} catch (Exception e) {

System.out.println("The server is down!");

}

}

}

class Language {

public static String AUTO\_DETECT = "auto";

public static String AFRIKAANS = "af";

public static String ALBANIAN = "sq";

public static String AMHARIC = "am";

public static String ARABIC = "ar";

public static String ARMENIAN = "hy";

public static String AZERBAIJANI = "az";

public static String BASQUE = "eu";

public static String BELARUSIAN = "be";

public static String BENGALI = "bn";

public static String BOSNIAN = "bs";

public static String BULGARIAN = "bg";

public static String CATALAN = "ca";

public static String CEBUANO = "ceb";

public static String CHICHEWA = "ny";

public static String CHINESE\_SIMPLIFIED = "zh-CN";

public static String CHINESE\_TRADITIONAL = "zh-TW";

public static String CORSICAN = "co";

public static String CROATIAN = "hr";

public static String CZECH = "cs";

public static String DANISH = "da";

public static String DUTCH = "nl";

public static String ENGLISH = "en";

public static String ESPERANTO = "eo";

public static String ESTONIAN = "et";

public static String FILIPINO = "tl";

public static String FINNISH = "fi";

public static String FRENCH = "fr";

public static String FRISIAN = "fy";

public static String GALICIAN = "gl";

public static String GEORGIAN = "ka";

public static String GERMAN = "de";

public static String GREEK = "el";

public static String GUJARATI = "gu";

public static String HAITIAN\_CREOLE = "ht";

public static String HAUSA = "ha";

public static String HAWAIIAN = "haw";

public static String HEBREW = "iw";

public static String HINDI = "hi";

public static String HMONG = "hmn";

public static String HUNGARIAN = "hu";

public static String ICELANDIC = "is";

public static String IGBO = "ig";

public static String INDONESIAN = "id";

public static String IRISH = "ga";

public static String ITALIAN = "it";

public static String JAPANESE = "ja";

public static String JAVANESE = "jw";

public static String KANNADA = "kn";

public static String KAZAKH = "kk";

public static String KHMER = "km";

public static String KOREAN = "ko";

public static String KURDISH\_KURMANJI = "ku";

public static String KYRGYZ = "ky";

public static String LAO = "lo";

public static String LATIN = "la";

public static String LATVIAN = "lv";

public static String LITHUANIAN = "lt";

public static String LUXEMBOURGISH = "lb";

public static String MACEDONIAN = "mk";

public static String MALAGASY = "mg";

public static String MALAY = "ms";

public static String MALAYALAM = "ml";

public static String MALTESE = "mt";

public static String MAORI = "mi";

public static String MARATHI = "mr";

public static String MONGOLIAN = "mn";

public static String MYANMAR\_BURMESE = "my";

public static String NEPALI = "ne";

public static String NORWEGIAN = "no";

public static String PASHTO = "ps";

public static String PERSIAN = "fa";

public static String POLISH = "pl";

public static String PORTUGUESE = "pt";

public static String PUNJABI = "pa";

public static String ROMANIAN = "ro";

public static String RUSSIAN = "ru";

public static String SAMOAN = "sm";

public static String SCOTS\_GAELIC = "gd";

public static String SERBIAN = "sr";

public static String SESOTHO = "st";

public static String SHONA = "sn";

public static String SINDHI = "sd";

public static String SINHALA = "si";

public static String SLOVAK = "sk";

public static String SLOVENIAN = "sl";

public static String SOMALI = "so";

public static String SPANISH = "es";

public static String SUNDANESE = "su";

public static String SWAHILI = "sw";

public static String SWEDISH = "sv";

public static String TAJIK = "tg";

public static String TAMIL = "ta";

public static String TELUGU = "te";

public static String THAI = "th";

public static String TURKISH = "tr";

public static String UKRAINIAN = "uk";

public static String URDU = "ur";

public static String UZBEK = "uz";

public static String VIETNAMESE = "vi";

public static String WELSH = "cy";

public static String XHOSA = "xh";

public static String YIDDISH = "yi";

public static String YORUBA = "yo";

public static String ZULU = "zu";

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package appservices;

import java.io.\*;

import java.net.ServerSocket;

import java.net.Socket;

/\*\*

\*

\* @author Admin

\*/

class S2Session extends Thread {

Socket conn;

public S2Session(Socket conn) {

this.conn = conn;

}

public void run() {

try {

System.out.println("Client "+ chatServer.clientNum +" is connected now!");

BufferedReader bufr = new BufferedReader(new InputStreamReader(conn.getInputStream()));

BufferedWriter bufw = new BufferedWriter(new OutputStreamWriter(conn.getOutputStream()));

String data = bufr.readLine();

while (!data.equals("bye")) {

bufw.write("Client " + TranslatorServer.clientNum + " : " + data);

bufw.newLine();

bufw.flush();

data = bufr.readLine();

}

conn.close();

}

catch (Exception e) {

e.printStackTrace();

}

}

}

public class chatServer {

/\*\*

\* @param args the command line arguments

\*/

static int clientNum = 0;

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

// TODO code application logic here

try {

ServerSocket chat = new ServerSocket(5472);

System.out.println("Chat Server is running now ...");

while (true) {

new S2Session(chat.accept()).start();

}

} catch (Exception e) {

System.out.println("The server is down!");

}

}

}

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package appservices;

/\*\*

\*

\* @author Admin

\*/

import java.io.\*;

import java.net.\*;

class getLastPosts extends Thread {

Socket con;

public getLastPosts(Socket con) {

this.con = con;

}

public void run() {

try {

//here what your server should do ....

} catch (Exception e) {

e.printStackTrace();

}

}

}

public class postsServer {

/\*\*

\* @param args the command line arguments

\*/

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

// TODO code application logic here

try {

ServerSocket posts = new ServerSocket(5555);

System.out.println("Posts Server is running now ...");

while (true) {

new getLastPosts(posts.accept()).start();

}

} catch (Exception e) {

System.out.println("The server is down!");

}

}

}