**LAB : 6**

**OBJECTIVE :**

Create an

1. Make application to login in HTTP server
2. Create mobile application which stores contacts in rms
3. Create application to sort number or characters using RMS
4. Create mobile application to search from records

**Requrements :**

(a) Windows PC (Windows 7/8/10) / Mac

(b) JDK 1.5

(c) Java Wireless Toolkit 2.5.2

**Implementation :**

1. Objective: Make application to login in HTTP server.

**login.java :**

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

import javax.microedition.io.\*;

import java.io.\*;

import java.lang.\*;

public class login extends MIDlet implements CommandListener {

public Form form1;

public Form form2;

public Command okCommand;

public Display display;

public HttpConnection ht=null;

public InputStream ist=null;

public StringItem st;

public TextField t1;

public TextField t2;

public StringBuffer buffer = new StringBuffer();

public TextBox access;

public login()

{

display=Display.getDisplay(this);

st=new StringItem(" "," Welcome");

t1=new TextField("UserName"," ",30,TextField.ANY);

t2=new TextField("Password"," ",30,TextField.PASSWORD);

form1=new Form("Login Here");

form2=new Form("Welcome");

okCommand=new Command("Login",Command.OK,1);

form1.addCommand(okCommand);

form1.setCommandListener(this);

form1.append(t1);

form1.append(t2);

}

public void startApp() {

display.setCurrent(form1);

}

public void pauseApp() {

}

public void destroyApp(boolean unconditional) {

notifyDestroyed();

}

public void commandAction(Command cmd,Displayable d)

{

if(cmd==okCommand)

{

try

{

// String url="http://192.168.5.19:8080/WebApplication7/index.jsp?t1=101&t2=aaa";

String url="http://127.0.0.1:12357/login/su1?pass=11&user=sudhanshu";

//ht=(HttpConnection)Connector.open("http://192.168.5.19:8080/WebApplication7/index.jsp");

ht=(HttpConnection)Connector.open(url);

ist=ht.openInputStream();

int chars;

while((chars = ist.read()) != -1){

buffer.append((char) chars);

}

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

access = new TextBox("Access Text", buffer.toString(), 1024, 0);

//form2.append(access);

display.setCurrent(access);

}

catch (Exception e){

form1.append(e.getMessage());

}

//finally{

//if(ist != null){

//ist.close();

//}}

}

}

}

**JSP Code :**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ page import="java.sql.\*" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Details</title>

</head>

<body>

<div style="padding:5px 400px; border:4px solid orange; border-radius:4px;">

<%

String url ="jdbc:mysql://localhost:3306/studentdetails";

String name ="root";

String pass ="1219";

String query="select \* from student";

int pass= Integer.parseInt(request.getParameter("pass"));

session.setAttribute("pass", pass);

Connection con = DriverManager.getConnection(url,name,pass);

Statement st=con.createStatement();

ResultSet rs = st.executeQuery(query);

int flag=0;

while(rs.next())

{

if(rs.getInt(1)==pass)

{

out.println("<b>Roll No:</b>"+rs.getInt(1)+"<br><b>Name is: </b>"+rs.getString(2)+"<br><b>Phone No: </b>"+rs.getInt(3)+"<br><b>Your father name is </b>"+rs.getString(4));

flag=1;

}

}

if(flag==0)

out.println("<b>Invalid roll Number please type correct Roll Number</b>");

st.close();

con.close();

%>

</div>

<fieldset style="display:flex; margin:auto">

<form action="su2">

New Phone no: <input type="number" name="phone" />

<input type="submit" value="update now"/>

</form>

</fieldset>

<br>

<br>

<br>

<fieldset>

<h3><b><i>for logout <a href="login.html"> click me</a></i></b></h3>

</fieldset>

</body>

</html>

Note: This app serves details of student after verifying username and passworsd. The problem which came is that it is serving html code instead parsing it.

**Output :**

|  |  |
| --- | --- |
|  |  |

1. Create mobile application which stores contacts in rms

**ContactNumber.java :**

import javax.microedition.rms.\*;

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

import java.io.\*;

public class ContactNumber extends MIDlet implements CommandListener {

private Display display;

private Alert alert;

private Form form;

private Command exit;

private Command start;

private RecordStore recordstore = null;

private RecordEnumeration recordEnumeration = null;

public ContactNumber() {

display = Display.getDisplay(this);

exit = new Command("Exit", Command.SCREEN, 1);

start = new Command("Start", Command.SCREEN, 1);

form = new Form("Mixed RecordEnumeration");

form.addCommand(exit);

form.addCommand(start);

form.setCommandListener(this);

}

public void startApp() {

display.setCurrent(form);

}

public void pauseApp() {

}

public void destroyApp( boolean unconditional ) {

}

public void commandAction(Command command, Displayable displayable) {

if (command == exit) {

destroyApp(true);

notifyDestroyed();

} else if (command == start) {

try {

recordstore = RecordStore.openRecordStore(

"myRecordStore", true );

} catch (Exception error) {

alert = new Alert("Error Creating",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

byte[] outputRecord;

String outputString[] = {"Arvin",

"Arti", "Vanshika"

};

int outputInteger[] = {983923103, 999989897, 987897897};

ByteArrayOutputStream outputStream =

new ByteArrayOutputStream();

DataOutputStream outputDataStream =

new DataOutputStream(outputStream);

for (int x = 0; x < 3; x++) {

outputDataStream.writeUTF(outputString[x]);

outputDataStream.writeInt(outputInteger[x]);

outputDataStream.flush();

outputRecord = outputStream.toByteArray();

recordstore.addRecord(outputRecord, 0,

outputRecord.length);

}

outputStream.reset();

outputStream.close();

outputDataStream.close();

} catch ( Exception error) {

alert = new Alert("Error Writing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

StringBuffer buffer = new StringBuffer();

byte[] byteInputData = new byte[300];

ByteArrayInputStream inputStream = new ByteArrayInputStream(byteInputData);

DataInputStream inputDataStream =

new DataInputStream(inputStream);

recordEnumeration = recordstore.enumerateRecords(

null, null, false);

while (recordEnumeration.hasNextElement()) {

recordstore.getRecord(recordEnumeration.nextRecordId(),

byteInputData, 0);

buffer.append(inputDataStream.readUTF());

buffer.append("\n");

buffer.append(inputDataStream.readInt());

buffer.append("\n");

alert = new Alert("Reading", buffer.toString(),

null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

inputStream.close();

} catch (Exception error) {

alert = new Alert("Error Reading",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

recordstore.closeRecordStore();

} catch (Exception error) {

alert = new Alert("Error Closing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

if (RecordStore.listRecordStores() != null) {

try {

RecordStore.deleteRecordStore("myRecordStore");

recordEnumeration.destroy();

} catch (Exception error) {

alert = new Alert("Error Removing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

}

}

}

}

**Output :**

|  |  |
| --- | --- |
|  |  |

1. Create application to sort number or characters using RMS

**SortExample.java :**

import javax.microedition.rms.\*;

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

import java.io.\*;

public class SortExample extends MIDlet implements CommandListener {

private Display display;

private Alert alert;

private Form form;

private Command exit;

private Command start;

private RecordStore recordstore = null;

private RecordEnumeration recordEnumeration = null;

private Comparator comparator = null;

public SortExample () {

display = Display.getDisplay(this);

exit = new Command("Exit", Command.SCREEN, 1);

start = new Command("Start", Command.SCREEN, 1);

form = new Form("Mixed RecordEnumeration", null);

form.addCommand(exit);

form.addCommand(start);

form.setCommandListener(this);

}

public void startApp() {

display.setCurrent(form);

}

public void pauseApp() {

}

public void destroyApp( boolean unconditional ) {

}

public void commandAction(Command command, Displayable displayable) {

if (command == exit) {

destroyApp(true);

notifyDestroyed();

} else if (command == start) {

try {

recordstore = RecordStore.openRecordStore(

"myRecordStore", true );

} catch (Exception error) {

alert = new Alert("Error Creating",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

String outputData[] = {"Mary", "Bob", "Adam"};

for (int x = 0; x < 3; x++) {

byte[] byteOutputData = outputData[x].getBytes();

recordstore.addRecord(byteOutputData, 0,

byteOutputData.length);

}

} catch ( Exception error) {

alert = new Alert("Error Writing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

StringBuffer buffer = new StringBuffer();

Comparator comparator = new Comparator();

recordEnumeration = recordstore.enumerateRecords(

null, comparator, false);

while (recordEnumeration.hasNextElement()) {

buffer.append(new String(recordEnumeration.nextRecord()));

buffer.append("\n");

}

alert = new Alert("Reading", buffer.toString() ,

null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

} catch (Exception error) {

alert = new Alert("Error Reading",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

recordstore.closeRecordStore();

} catch (Exception error) {

alert = new Alert("Error Closing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

if (RecordStore.listRecordStores() != null) {

try {

RecordStore.deleteRecordStore("myRecordStore");

recordEnumeration.destroy();

} catch (Exception error) {

alert = new Alert("Error Removing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

}

}

}

}

class Comparator implements RecordComparator {

public int compare(byte[] record1, byte[] record2) {

String string1 = new String(record1),

string2 = new String(record2);

int comparison = string1.compareTo(string2);

if (comparison == 0)

return RecordComparator.EQUIVALENT;

else if (comparison < 0)

return RecordComparator.PRECEDES;

else

return RecordComparator.FOLLOWS;

}

}

Note: In this application we need to sort names consisting of Adam, Bob and Mary

**Output :**

|  |  |
| --- | --- |
|  |  |

1. Create mobile application to search from records.

(Note: In this application we need to find or search Samay from already sorted list application which consist name having Arvin, Samay, Tanmay)

**SearchExample.java :**

import javax.microedition.rms.\*;

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

import java.io.\*;

public class SearchExample extends MIDlet implements CommandListener {

private Display display;

private Alert alert;

private Form form;

private Command exit;

private Command start;

private RecordStore recordstore = null;

private RecordEnumeration recordEnumeration = null;

private Filter filter = null;

public SearchExample () {

display = Display.getDisplay(this);

exit = new Command("Exit", Command.SCREEN, 1);

start = new Command("Start", Command.SCREEN, 1);

form = new Form("Mixed RecordEnumeration", null);

form.addCommand(exit);

form.addCommand(start);

form.setCommandListener(this);

}

public void startApp() {

display.setCurrent(form);

}

public void pauseApp() {

}

public void destroyApp( boolean unconditional ) {

}

public void commandAction(Command command, Displayable displayable) {

if (command == exit) {

destroyApp(true);

notifyDestroyed();

} else if (command == start) {

try {

recordstore = RecordStore.openRecordStore(

"myRecordStore", true );

} catch (Exception error) {

alert = new Alert("Error Creating",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

String outputData[] = {"Arvin", "Samay", "Tanmay"};

for (int x = 0 ; x < 3; x++) {

byte[] byteOutputData = outputData[x].getBytes();

recordstore.addRecord(byteOutputData, 0,

byteOutputData.length);

}

} catch ( Exception error) {

alert = new Alert("Error Writing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

filter = new Filter("Samay");

recordEnumeration = recordstore.enumerateRecords(

filter, null, false);

if (recordEnumeration.numRecords() > 0) {

String string = new String(recordEnumeration.nextRecord());

alert = new Alert("Reading", string,

null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

} catch (Exception error) {

alert = new Alert("Error Reading",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

try {

recordstore.closeRecordStore();

} catch (Exception error) {

alert = new Alert("Error Closing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

if (RecordStore.listRecordStores() != null) {

try {

RecordStore.deleteRecordStore("myRecordStore");

recordEnumeration.destroy();

filter.filterClose();

} catch (Exception error) {

alert = new Alert("Error Removing",

error.toString(), null, AlertType.WARNING);

alert.setTimeout(Alert.FOREVER);

display.setCurrent(alert);

}

}

}

}

}

class Filter implements RecordFilter {

private String search = null;

private ByteArrayInputStream inputstream = null;

private DataInputStream datainputstream = null;

public Filter(String search) {

this.search = search.toLowerCase();

}

public boolean matches(byte[] suspect) {

String string = new String(suspect).toLowerCase();

if (string != null && string.indexOf(search) != -1)

return true;

else

return false;

}

public void filterClose() {

try {

if (inputstream != null) {

inputstream.close();

}

if (datainputstream != null) {

datainputstream.close();

}

} catch ( Exception error) {

}

}

}

**Output :**

|  |  |
| --- | --- |
|  |  |