L A M P I R A N

1. Listing Program index.wml

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<template>
<do type="prev" label="back">
<prev/>
</do>
</template>
<card id="index" title="Welcome">
<p align="left">
1. <a href="what.wml">What's J2ME</a><br/>
2. <a href="support.wml">Products Support</a><br/>
3. <a href="down.wml">Download</a><br/>
4. <a href="reg.wml">Registration</a><br/>
5. <a href="help.wml">Help</a><br/>
</p>
</card>
</wml>
```

2. Listing Program what.wml

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<template>
<do type="prev" label="back">
<prev/>
</do>
</template>
<card id="what" title="J2ME">
<p align="left">
The Java 2 Platform, Micro Edition (J2ME)
is the Java platform for consumer and embedded
devices such as mobile phones, PDAs,
TV set-top boxes, in-vehicle telematics systems,
and a broad range of embedded devices.<br/>
With J2ME, applications are written once for a wide
range of devices, are downloaded dynamically,
and leverage each device's native
capabilities.<br/>
The J2ME platform is deployed on millions
of devices — from mobile phones, to PDAs,
to automotive devices — supported by leading
Java technology tools vendors, and used
by companies worldwide. In short, it is the
```

```
platform of choice for today's consumer
and embedded devices.<br/>
</p>
</card>
</wml>
```

3. Listing Program support.wml

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<template>
<do type="prev" label="back">
<prev/>
</do>
</template>
<card id="support" title="Products">
<p align="left">
<b>Motorola:</b>
<table columns="3">
<tr>
<td>A388</td>
<td>A820 </td>
<td>V60i </td>
</tr>
<tr>
<td>V66i</td>
<td>T280i </td>
<td>T720</td>
</tr>
</table>
<b>Nokia:</b>
<table columns="3">
<tr>
<td>3410</td>
<td>3510i</td>
<td>3530</td>
</tr>
<tr>
<td>3590</td>
<td>3650</td>
<td>5100</td>
</tr>
<tr>
<td>6100</td>
<td>6200i</td>
<td>6310i</td>
</tr>
<tr>
<td>6610</td>
<td>6650</td>
<td>6800</td>
</tr>
<tr>
```

```
<td>7210</td>
<td>7250</td>
<td>7650</td>
</tr>
<tr>
<td>8910i</td>
<td>9210</td>
<td>9210i</td>
</tr>
<tr>
<td>9290</td>
</tr>
</table>
<b>Samsung</b>
<table columns="1">
<tr>
<td>SGH-S100 </td>
</tr>
</table>
<b>Siemens</b>
<table columns="3">
<tr>
<td>C55 </td>
<td>M46</td>
<td>M50</td>
</tr>
<tr>
<td>SL42</td>
<td>SL45i</td>
<td>S55</td>
</tr>
</table>
<b>Sony Ericsson</b>
<table columns="1">
<tr>
<td>P800 </td>
</tr>
</table>
</p>
</card>
</wml>
</wml>
```

4. Listing Program download.wml

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="download" title="Try">
<p align="left">
<b>Member Area</b><br/><br/>
<do type="accept" label="login">
<go method="post" href="cek.php">
<postfield name="user" value="$(user)"/>
```

```
<postfield name="pass" value="$(pass)"/>
</go>
</do>
Username:<br/>
<input name="user" type="text" size="6" maxlength="6" /><br/>
Password:<br/>
<input name="pass"  type="password" size="6" maxlength="6" /><br/>
<b>Free Download</b><br/>
TimerDemo<br/>
<a href="http://202.43.254.3/ta/TimerDemo.jad">[download]</a><br/>
</p>
</card>
</wml>
```

5. Listing Program reg.wml

```
<?xml version="1.0"?>
<!DOCTYPE     wml     PUBLIC     "-//WAPFORUM//DTD     WML     1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="form" title="Registration">
<p align="left">
<do type="accept" label="submit">
<go method="post" href="queform1.php">
<postfield name="fname" value="$(fname)"/>
<postfield name="lname" value="$(lname)"/>
<postfield name="region" value="$(region)"/>
<postfield name="regcode" value="$(regcode)"/>
<postfield name="nocell" value="$(nocell)"/>
<postfield name="email" value="$(email)"/>
<postfield name="gender" value="$(gender)"/>
<postfield name="age" value="$(age)"/>
<postfield name="occup" value="$(occup)"/>
<postfield name="user" value="$(user)"/>
<postfield name="pass" value="$(pass)"/>
<postfield name="repass" value="$(repass)"/>
</go>
</do>
<b>Profile Info</b><br/>
First Name:<br/>
      <input name="fname" type="text" maxlength="18"/><br/>
Last Name:<br/>
      <input name="lname"  type="text" maxlength="18"/><br/>
Country/Region:<br/>
      <input name="region"  type="text" maxlength="18"/><br/>
Country/Region Code:<br/>
      <input        name="regcode"        type="text"        size="4"
maxlength="3"/><br/>
No. Cell:<br/>
      <input name="nocell" type="text" maxlength="18"/><br/>
Email Address:<br/>
      <input name="email" type="text" maxlength="36"/><br/>
Gender: <small>(m/f)</small><br/>
      <input      name="gender"            type="text"      size="2"
maxlength="1"/><br/>
```

```
Age: <br/>
      <input name="age"  type="text" size="4" maxlength="3"/><br/>
Occupation:<br/>
      <input name="occup"  type="text" maxlength="18"/><br/><br/>
<b>Account Info</b><br/>
Username:<br/>
<input name="user" type="text" size="6" maxlength="6"/><br/>
Password:<br/>
<input name="pass" type="password" size="6" maxlength="6"/><br/>
Re-Password:<br/>
<input name="repass" type="password" size="6" maxlength="6"/><br/>
</p>
</card>
</wml>
```

6. Listing Program help.wml

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<template>
<do type="prev" label="back">
<prev/>
</do>
</template>
<card id="help" title="Instructions">
<p align="left">
1. If you don't have an account, Please register first. It's
<b>free</b><br/>
2. You can get Java App. in Download's Page<br/>
3. Click the button to download Java App.<br/>
4. Download and Installation starts automatically<br/>
5. If you failed with GPRS, use CSD Connection.<br/>
</p>
</card>
</wml>
```

7. Listing Program queform1.php

```
<?
Header('Content-type:text/vnd.wap.wml');
echo ('<?xml version="1.0"?>');
echo ('<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">');
?>
<wml>
<card id="verform1" title="Registration">
<p align="left">
<do type="accept" label="submit">
<go method="post" href="queform2.php">
<postfield name="fname" value="$(fname)"/>
<postfield name="lname" value="$(lname)"/>
<postfield name="region" value="$(region)"/>
<postfield name="regcode" value="$(regcode)"/>
<postfield name="nocell" value="$(nocell)"/>
```

```
<postfield name="email" value="$(email)"/>
<postfield name="gender" value="$(gender)"/>
<postfield name="age" value="$(age)"/>
<postfield name="occup" value="$(occup)"/>
<postfield name="user" value="$(user)"/>
<postfield name="pass" value="$(pass)"/>
<postfield name="repass" value="$(repass)"/>
</go>
</do>
<?
$hostname="localhost";
$username="";
$password="";
$conn = mysql_connect($hostname, $username, $password);
if(!$conn){
echo "Could not connect to database server - please try later.";
}
mysql_select_db("ta", $conn);
$tanggal=date("D, d F Y H:i:s A");
$result = mysql_query("select * from user_acc where user_name =
('$user')", $conn);
if ($fname != null && $lname != null && $region != null && $regcode
!= null && $nocell != null && $email != null && $gender != null &&
$age != null && $occup != null && $user != null && $pass != null &&
$repass != null && ($pass == $repass) && (mysql_num_rows($result) <=
0)){
$isipro="insert into user_pro(user_fname, user_lname, user_region,
user_regcode, user_nocell, user_email, user_gender, user_age,
user_occup) values ('$fname', '$lname', '$region', '$regcode',
'$nocell', '$email', '$gender', '$age', '$occup')";
$hasilpro=mysql_query($isipro);
}
else{
if ($fname == null){
?>
      First Name:<br/>
      <input name="fname" type="text" maxlength="18"/><br/>
<?
}
if ($lname == null){
?>
      Last Name:<br/>
      <input name="lname"  type="text" maxlength="18"/><br/>
<?
}
if ($region == null){
?>
      Region:<br/>
      <input name="region"  type="text" maxlength="18"/><br/>
<?
}
if ($regcode == null){
?>
      Country/Region Code:<br/>
```

```
      <input name="regcode" type="text" size="4"
maxlength="3"/><br/>
<?
}
if ($nocell == null){
?>
      No. Cell:<br/>
      <input name="nocell" type="text" maxlength="18"/><br/>
<?
}
if ($email == null){
?>
      Email Address:<br/>
      <input name="email" type="text" maxlength="36"/><br/>
<?
}
if ($gender == null){
?>
      Gender:<small>(m/f)</small><br/>
      <input name="gender"  type="text" size="2"
maxlength="1"/><br/>
<?
}
if ($age == null){
?>
      Age:<br/>
      <input name="age"  type="text" size="4" maxlength="3"/><br/>
<?
}
if ($occup == null){
?>
      Occupation:<br/>
      <input name="occup"  type="text" maxlength="18"/><br/><br/>
<?
}
}
if ($fname != null && $lname != null && $region != null && $regcode
!= null && $nocell != null && $email != null && $gender != null &&
$age != null && $occup != null && $user != null && $pass != null &&
$repass != null && ($pass == $repass) && (mysql_num_rows($result) <=
0)){
$isiacc="insert into user_acc(user_name, user_pass, user_repass,
user_date) values ('$user', '$pass', '$repass', '$tanggal')";
$hasilacc=mysql_query($isiacc);
      if(!$hasilacc){
      echo"query failed<br/>";
      }
      else{
      echo"query success<br/>";
?>
<a href="index.wml"> Go Home </a>
<?
}
}
```

```
elseif (mysql_num_rows($result) > 0){
?>
<small><b>Type other Username</b></small><br/>
<input name="user" type="text" size="6"/><br/>
<?
}
elseif ($pass != $repass){
?>
<small>Password entries do not match.</small><br/>
<small><b>Type Password</b></small><br/>
<input name="pass"  type="password" size="6"/><br/>
<small><b>Type Re-password</b></small><br/>
<input name="repass"  type="password" size="6"/><br/>
<?
}
else{
if ($user == null){
?>
      <small><b>Type Username</b></small><br/>
      <input name="user" type="text" size="6"/><br/>
<?
}
if ($pass == null){
?>
      <small><b>Type Password</b></small><br/>
      <input name="pass"  type="text" size="6"/><br/>
<?
}
if ($repass == null){
?>
      <small><b>Type Re-Password</b></small><br/>
      <input name="repass"  type="text" size="6"/><br/>
<?
}
}
?>
</p>
</card>
</wml>
```

8. Listing Program queform2.php

```
<?
Header('Content-type:text/vnd.wap.wml');
echo ('<?xml version="1.0"?>');
echo ('<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">');
?>
<wml>
<card id="verform1" title="Registration">
<p align="left">
<do type="accept" label="submit">
<go method="post" href="queform1.php">
<postfield name="fname" value="$(fname)"/>
<postfield name="lname" value="$(lname)"/>
```

```
<postfield name="region" value="$(region)"/>
<postfield name="regcode" value="$(regcode)"/>
<postfield name="nocell" value="$(nocell)"/>
<postfield name="email" value="$(email)"/>
<postfield name="gender" value="$(gender)"/>
<postfield name="age" value="$(age)"/>
<postfield name="occup" value="$(occup)"/>
<postfield name="user" value="$(user)"/>
<postfield name="pass" value="$(pass)"/>
<postfield name="repass" value="$(repass)"/>
</go>
</do>
<?
$hostname="localhost";
$username="";
$password="";
$conn = mysql_connect($hostname, $username, $password);
if(!$conn){
echo "Could not connect to database server - please try later.";
}
mysql_select_db("ta", $conn);
$tanggal=date("D, d F Y H:i:s A");
$result = mysql_query("select * from user_acc where user_name =
('$user')", $conn);
if ($fname != null && $lname != null && $region != null && $regcode
!= null && $nocell != null && $email != null && $gender != null &&
$age != null && $occup != null && $user != null && $pass != null &&
$repass != null && ($pass == $repass) && (mysql_num_rows($result) <=
0)){
$isipro="insert into user_pro(user_fname, user_lname, user_region,
user_regcode, user_nocell, user_email, user_gender, user_age,
user_occup) values ('$fname', '$lname', '$region', '$regcode',
'$nocell', '$email', '$gender', '$age', '$occup')";
$hasilpro=mysql_query($isipro);
}
else{
if ($fname == null){
?>
      First Name:<br/>
      <input name="fname" type="text" maxlength="18"/><br/>
<?
}
if ($lname == null){
?>
      Last Name:<br/>
      <input name="lname"  type="text" maxlength="18"/><br/>
<?
}
if ($region == null){
?>
      Region:<br/>
      <input name="region"  type="text" maxlength="18"/><br/>
<?
}
```

```
if ($regcode == null){
?>
      Country/Region Code:<br/>
      <input name="regcode" type="text" size="4"
maxlength="3"/><br/>
<?
}
if ($nocell == null){
?>
      No. Cell:<br/>
      <input name="nocell" type="text" maxlength="18"/><br/>
<?
}
if ($email == null){
?>
      Email Address:<br/>
      <input name="email" type="text" maxlength="36"/><br/>
<?
}
if ($gender == null){
?>
      Gender:<small>(m/f)</small><br/>
      <input name="gender"  type="text" size="2"
maxlength="1"/><br/>
<?
}
if ($age == null){
?>
      Age:<br/>
      <input name="age"  type="text" size="4" maxlength="3"/><br/>
<?
}
if ($occup == null){
?>
      Occupation:<br/>
      <input name="occup"  type="text" maxlength="18"/><br/><br/>
<?
}
}
if ($fname != null && $lname != null && $region != null && $regcode
!= null && $nocell != null && $email != null && $gender != null &&
$age != null && $occup != null && $user != null && $pass != null &&
$repass != null && ($pass == $repass) && (mysql_num_rows($result) <=
0)){
$isiacc="insert into user_acc(user_name, user_pass, user_repass,
user_date) values ('$user', '$pass', '$repass', '$tanggal')";
$hasilacc=mysql_query($isiacc);
      if(!$hasilacc){
      echo"query failed<br/>";
      }
      else{
      echo"query success<br/>";
?>
<a href="index.wml"> Go Home </a>
```

```
<?
}
}
elseif (mysql_num_rows($result) > 0){
?>
<small><b>Type other Username</b></small><br/>
<input name="user" type="text" size="6"/><br/>
<?
}
elseif ($pass != $repass){
?>
<small>Password entries do not match.</small><br/>
<small><b>Type Password</b></small><br/>
<input name="pass"  type="password" size="6"/><br/>
<small><b>Type Re-password</b></small><br/>
<input name="repass"  type="password" size="6"/><br/>
<?
}
else{
if ($user == null){
?>
      <small><b>Type Username</b></small><br/>
      <input name="user" type="text" size="6"/><br/>
<?
}
if ($pass == null){
?>
      <small><b>Type Password</b></small><br/>
      <input name="pass"  type="text" size="6"/><br/>
<?
}
if ($repass == null){
?>
      <small><b>Type Re-Password</b></small><br/>
      <input name="repass"  type="text" size="6"/><br/>
<?
}
}
?>
</p>
</card>
</wml>
```

9. Listing Program cek.php

```
<?
Header('Content-type:text/vnd.wap.wml');
echo ('<?xml version="1.0"?>');
echo   ('<!DOCTYPE   wml   PUBLIC   "-//WAPFORUM//DTD   WML   1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">');
?>
<wml>
<card id="koneksi" title="Download">
<p align="left">
<?
```

```
$hostname="localhost";
$username="";
$password="";
$conn = mysql_connect($hostname, $username, $password);
mysql_select_db("ta", $conn);
$result = mysql_query("select * from user_acc where user_name =
('$user') and user_pass = ('$pass')", $conn);
if (mysql_num_rows($result) > 0)
{?>
Welcome $user, <br/>
1. To-Do List<br/>
      It allows the user to create, modify and delete tasks stored
in the device memory.<br/>
      <a
href="http://202.43.254.3/ta/todo.jad">[download]</a><br/><br/>
2. Address Book<br/>
      It allows the user to store address, email, and more phone
number, An excellent utility for people on the move<br/>
      <a
href="http://202.43.254.3/ta/adbok.jad">[download]</a><br/><br/><br/
>
3. Scheduler<br/>
      Scheduler is a basic scheduling utility allowing you to enter
and store events in a calendar<br/>
      <a
href="http://202.43.254.3/ta/sched.jad">[download]</a><br/><br/>
<?
}
else
echo "You are not Authorized";
?>
</p>
</card>
</wml>
```

10. Listing Program "To Do"

```
import javax.microedition.rms.*;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;

public class todo extends MIDlet
implements CommandListener {
Display display = null;
private Command backCmd = new Command("Back", Command.OK, 2);
private Command exitCmd = new Command("Exit", Command.STOP, 2);
private Command addCmd = new Command("Add", Command.OK, 1);
private Command viewCmd = new Command("View", Command.OK, 1);
private Command editCmd = new Command("Edit", Command.OK, 1);
private Command eraseCmd = new Command("Erase", Command.OK, 1);
private Command saveCmd = new Command("Save", Command.OK, 1);
private Command yesCmd = new Command("Yes", Command.OK, 1);
private Command replaceCmd = new Command("Replace", Command.OK, 1);
private Form f;
private TextField t;
```

```
private Alert a;
private RecordStore myDb;
private ChoiceGroup dt;
private StringItem si;

public todo() {
}

public void startApp()
throws MIDletStateChangeException {
      display = Display.getDisplay(this);
      a     =     new Alert("To-Do List");
      a.setType(AlertType.WARNING);
      mulaiTampilDb();
}

public void pauseApp() {
      display = null;
}

public void destroyApp(boolean unconditional) {
      notifyDestroyed();
}

public void mulaiTambahDb() {
      display     =     Display.getDisplay(this);
      f = new Form("To-Do List");
      t           =     new
      TextField("Subject:",null,32,TextField.ANY);
      f.append(t);
      f.addCommand(saveCmd);
      f.addCommand(backCmd);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void tambahDb() {
      display = Display.getDisplay(this);
      try {
            myDb = RecordStore.openRecordStore(t.getString(),true);
            a.setString("Notes " + t.getString() + " Saved");
            display.setCurrent(a);
      }
      catch(RecordStoreException e) {
            a.setString("Internal Error");
            display.setCurrent(a);
      }
}

public void mulaiTampilDb() {
      if(RecordStore.listRecordStores() == null ) {
            tidakadaDb();
      }
      if(RecordStore.listRecordStores() != null ) {
```

```
            adaDb();
      }
}

public void tidakadaDb() {
      display = Display.getDisplay(this);
      f = new Form("To-Do List");
      f.append("(No Notes)");
      f.addCommand(addCmd);
      f.addCommand(exitCmd);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void adaDb() {
      display     =     Display.getDisplay(this);
      f = new Form("To-Do List");
      dt          =     new ChoiceGroup("Notes",Choice.EXCLUSIVE);
      String allDbs[] = RecordStore.listRecordStores();
      for(int i = 0; i < allDbs.length; i++){
            dt.append(allDbs[i],null);
      }
      f.append(dt);
      f.addCommand(exitCmd);
      f.addCommand(backCmd);
      f.addCommand(addCmd);
      f.addCommand(viewCmd);
      f.addCommand(editCmd);
      f.addCommand(eraseCmd);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void tampilDb() {
      String dbName = dt.getString(dt.getSelectedIndex());
      f = new Form("To-Do List");
      si = new StringItem("",dbName);
      f.append(si);
      f.addCommand(backCmd);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void mulaiHapusDb() {
      String dbName = dt.getString(dt.getSelectedIndex());
      f = new Form("To-Do List");
      f.append("Delete " + dbName+ "?");
      f.addCommand(yesCmd);
      f.addCommand(backCmd);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void hapusDb() {
```

```
        try {
              display     = Display.getDisplay(this);
              String allDbs[] = RecordStore.listRecordStores();
              for(int i = 0; i < allDbs.length; i++){
                    dt.append(allDbs[i],null);
              }
              String dbName = dt.getString(dt.getSelectedIndex());
              dt.delete(dt.getSelectedIndex());
              RecordStore.deleteRecordStore(dbName);
              a.setString("Notes " + dbName+ "Erased");
              display.setCurrent(a);
        }
        catch(RecordStoreNotFoundException e) {
        }
        catch(RecordStoreException e) {
              a.setString("Internal Error");
              display.setCurrent(a);
        }
        }

  public void mulaiEditDb() {
        display     = Display.getDisplay(this);
        f = new Form("To-Do List");
        String allDbs[] = RecordStore.listRecordStores();
        for(int i = 0; i < allDbs.length; i++){
        dt.append(allDbs[i],null);
        }
        String dbName = dt.getString(dt.getSelectedIndex());
        t           =     new
        TextField("Subject:",dbName,32,TextField.ANY);
        f.append(t);
        f.addCommand(replaceCmd);
        f.addCommand(backCmd);
        f.setCommandListener(this);
        display.setCurrent(f);
        }

  public void commandAction(Command c, Displayable d) {
        String label = c.getLabel();
        if (label.equals("Exit")) {
              destroyApp(true);
        }
        if (label.equals("Back")) {
               mulaiTampilDb();
        }
        if (label.equals("Add"))  {
              mulaiTambahDb();
        }
        if (label.equals("View")) {
              tampilDb();
        }
        if (label.equals("Edit")) {
              mulaiEditDb();
        }
```

```
      if (label.equals("Erase")){
            mulaiHapusDb();
      }
      if (label.equals("Save")) {
            tambahDb();
            mulaiTampilDb();
      }
      if (label.equals("Yes")) {
            hapusDb();
            mulaiTampilDb();
      }
      if (label.equals("Replace")) {
            hapusDb();
            tambahDb();
            mulaiTampilDb();
      }
      }
      }
```

11. Listing Program "Address Book"

```
import java.io.*;
import javax.microedition.rms.*;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;


public class adbok extends MIDlet implements CommandListener {

Display display = null;
List menu = null;
TextBox input = null;

static final Command backCommand = new Command("Back", Command.BACK,
0);
static final Command mainMenuCommand = new Command("Main",
Command.SCREEN, 1);
static final Command exitCommand = new Command("Exit", Command.STOP,
2);

private Command AddCmd = new Command("Add",Command.OK,1);
private Command SaveCmd = new Command("Save",Command.OK,1);
private Command DetailCmd = new Command("Detail",Command.OK,1);
private Command EditCmd = new Command("Edit",Command.OK,1);
private Command EraseCmd = new Command("Erase",Command.OK,1);
private Command ReplaceCmd = new Command("Replace",Command.OK,1);
private Command YesCmd = new Command("Yes",Command.OK,1);
private Form f, f2, f_mTDb, f_tDbD, f_mEDb, f_mEDbD;
private TextField t, t2, t_mEDbD;
private TextField fname, lname, email, phone;
private TextField mobile, fax, street, zip, city, country;
private RecordStore DbAdBok;
private Alert a;
private ChoiceGroup dt, cg_mTDb, cg_mEDb;
private StringItem sifname, silname, siemail, siphone;
```

```
private StringItem simobile, sifax, sistreet, sizip, sicity,
                   sicountry;
String currentMenu = null;

public adbok() {
}

public void startApp()
throws MIDletStateChangeException {
      display = Display.getDisplay(this);
      a = new Alert("Address Book");
      a.setType(AlertType.WARNING);
      menu = new List("Address Book", Choice.IMPLICIT);
      menu.append("Add", null);
      menu.append("View", null);
      menu.append("Edit", null);
      menu.append("Erase", null);
      menu.addCommand(exitCommand);
      menu.setCommandListener(this);
      mainMenu();
}

public void pauseApp() {
      display = null;
      menu = null;
      input = null;
}

public void destroyApp(boolean unconditional) {
      notifyDestroyed();
}

void mainMenu() {
      display.setCurrent(menu);
      currentMenu = "Main";
}

public void mulaiTambahDb(){
      display = Display.getDisplay(this);
      f = new Form("Add");
      t = new TextField("Add Entry:",null,32,TextField.ANY);
      f.append(t);
      f.addCommand(AddCmd);
      f.addCommand(backCommand);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void tambahDb(){
try{
      DbAdBok = RecordStore.openRecordStore(t.getString(),true);
      a.setString("Entry" + t.getString()+ "Saved");
      display.setCurrent(a);
      bukaTambahDbData();
```

```
}
catch(RecordStoreException e) {
      a.setString("Entry Error ");
      display.setCurrent(a);
}
}

public void bukaTambahDbData(){
      try{
       DbAdBok = RecordStore.openRecordStore(t.getString(),false);
       mulaiTambahDbData();
      }
      catch(RecordStoreNotFoundException e){
      }
      catch(RecordStoreException e){
      }
}

public void mulaiTambahDbData(){
f2   = new Form("Entry " + t.getString());
lname = new TextField("Last Name:", null, 32, TextField.ANY);
fname = new TextField("First Name:", null, 32, TextField.ANY);
email = new TextField("E-Mail:", null, 32, TextField.EMAILADDR);
phone = new TextField("Phone:", null, 32, TextField.PHONENUMBER);
mobile = new TextField("Mobile:", null, 32, TextField.PHONENUMBER);
fax   = new TextField("Fax:", null, 32, TextField.PHONENUMBER);
street = new TextField("Street:", null, 32, TextField.ANY);
zip   = new TextField("Zip:", null, 32, TextField.NUMERIC);
city  = new TextField("City:", null, 32, TextField.ANY);
country = new TextField("Country:", null, 32, TextField.ANY);
f2.append(lname); f2.append(fname);
f2.append(email); f2.append(phone);
f2.append(mobile); f2.append(fax);
f2.append(street); f2.append(zip);
f2.append(city); f2.append(country);
f2.addCommand(SaveCmd);
f2.setCommandListener(this);
display.setCurrent(f2);
}

public void tambahDbData(){
try{
      ByteArrayOutputStream baos = new ByteArrayOutputStream();
      DataOutputStream dos = new DataOutputStream(baos);
      dos.writeUTF(lname.getString());
      byte[] blname = baos.toByteArray();
      DbAdBok.addRecord(blname, 0, blname.length);

      dos.writeUTF(fname.getString());
      byte[] bfname = baos.toByteArray();
      DbAdBok.addRecord(bfname, 0, bfname.length);

      dos.writeUTF(email.getString());
      byte[] bemail = baos.toByteArray();
```

```
        DbAdBok.addRecord(bemail, 0, bemail.length);

        dos.writeUTF(phone.getString());
        byte[] bphone = baos.toByteArray();
        DbAdBok.addRecord(bphone, 0, bphone.length);

        dos.writeUTF(mobile.getString());
        byte[] bmobile = baos.toByteArray();
        DbAdBok.addRecord(bmobile, 0, bmobile.length);

        dos.writeUTF(fax.getString());
        byte[] bfax = baos.toByteArray();
        DbAdBok.addRecord(bfax, 0, bfax.length);

        dos.writeUTF(street.getString());
        byte[] bstreet = baos.toByteArray();
        DbAdBok.addRecord(bstreet, 0, bstreet.length);

        dos.writeUTF(zip.getString());
        byte[] bzip = baos.toByteArray();
        DbAdBok.addRecord(bzip, 0, bzip.length);

        dos.writeUTF(city.getString());
        byte[] bcity = baos.toByteArray();
        DbAdBok.addRecord(bcity, 0, bcity.length);

        dos.writeUTF(country.getString());
        byte[] bcountry = baos.toByteArray();
        DbAdBok.addRecord(bcountry, 0, bcountry.length);

        mainMenu();
        }
        catch(IOException e){
        }
        catch(RecordStoreFullException e){
        }
        catch(RecordStoreNotOpenException e){
        }
        catch(RecordStoreException e){
        }
}

public void mulaiTampilDb(){
if(RecordStore.listRecordStores() == null ){
        f_mTDb = new Form("View");
        f_mTDb.append("(Empty)");
        f_mTDb.addCommand(backCommand);
        f_mTDb.setCommandListener(this);
        display.setCurrent(f_mTDb);
}
if(RecordStore.listRecordStores() != null ){
        f_mTDb = new Form("View");
        cg_mTDb =   new ChoiceGroup("Select
                    Entry:",ChoiceGroup.EXCLUSIVE);
```

```
      String allDbs[] = RecordStore.listRecordStores();
      for(int i = 0; i < allDbs.length; i++){
                  cg_mTDb.append(allDbs[i],null);
      }
      f_mTDb.append(cg_mTDb);
      f_mTDb.addCommand(DetailCmd);
      f_mTDb.addCommand(backCommand);
      f_mTDb.setCommandListener(this);
      display.setCurrent(f_mTDb);
}
}

public void bukaTampilDb(){
try{
      String dbName = cg_mTDb.getString(cg_mTDb.getSelectedIndex());
      DbAdBok = RecordStore.openRecordStore(dbName,false);
      tampilDbData();
}
catch(RecordStoreNotFoundException e){
}
catch(RecordStoreException e){
}
}



public void tampilDbData(){
try{
      f_tDbD   = new Form("Detail ");
      f_tDbD.addCommand(backCommand);
      f_tDbD.setCommandListener(this);

      ByteArrayInputStream bais;
      DataInputStream dis;
      String inlname;   String infname;
      String inemail; String inphone;
      String inmobile; String infax;
      String instreet; String inzip;
      String incity;    String incountry;

      int recId;

      RecordEnumeration enum =
      DbAdBok.enumerateRecords(null,null,false);
      recId = enum.nextRecordId();
      bais = new ByteArrayInputStream(DbAdBok.getRecord(recId));
      dis  = new DataInputStream(bais);

      inlname     = dis.readUTF();
      silname     = new StringItem("Last Name:",inlname);

      infname     = dis.readUTF();
      sifname     = new StringItem("First Name:",infname);
```

```
        inemail           = dis.readUTF();
        siemail     = new StringItem("Email:",inemail);

        inphone     = dis.readUTF();
        siphone     = new StringItem("Phone:",inphone);


        inmobile    = dis.readUTF();
        simobile    = new StringItem("Mobile:",inmobile);

        infax       = dis.readUTF();
        sifax       = new StringItem("Fax:",infax);

        instreet    = dis.readUTF();
        sistreet    = new StringItem("Street:",instreet);

        inzip       = dis.readUTF();
        sizip             = new StringItem("Zip:",inzip);

        incity      = dis.readUTF();
        sicity      = new StringItem("City:",incity);

        incountry   = dis.readUTF();
        sicountry   = new StringItem("Country:",incountry);

        f_tDbD.append(silname);       f_tDbD.append(sifname);
        f_tDbD.append(siemail);       f_tDbD.append(siphone);
        f_tDbD.append(simobile);      f_tDbD.append(sifax);
        f_tDbD.append(sistreet);      f_tDbD.append(sizip);
        f_tDbD.append(sicity);        f_tDbD.append(sicountry);

        display.setCurrent(f_tDbD);
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
}

public void mulaiEditDb(){
        if(RecordStore.listRecordStores() == null ){
              f_mEDb = new Form("Edit");
              f_mEDb.append("(empty)");
              f_mEDb.addCommand(backCommand);
              f_mEDb.setCommandListener(this);
              display.setCurrent(f_mEDb);
        }
        if(RecordStore.listRecordStores() != null ){
              f_mEDb = new Form("Edit");
              cg_mEDb = new ChoiceGroup("Select
                        Entry:",ChoiceGroup.EXCLUSIVE);
              String allDbs[] = RecordStore.listRecordStores();
```

```
            for(int i = 0; i < allDbs.length; i++){
                  cg_mEDb.append(allDbs[i],null);
            }
            f_mEDb.append(cg_mEDb);
            f_mEDb.addCommand(EditCmd);
            f_mEDb.addCommand(backCommand);
            f_mEDb.setCommandListener(this);
            display.setCurrent(f_mEDb);
      }
      }

public void bukaEditDb(){
try{
      String dbName = cg_mEDb.getString(cg_mEDb.getSelectedIndex());
      DbAdBok = RecordStore.openRecordStore(dbName,false);
      mulaiEditDbData();
}
catch(RecordStoreNotFoundException e){
}
catch(RecordStoreException e){
}
}



public void mulaiEditDbData(){
try{
      f_mEDbD   = new Form("Edit");
      f_mEDbD.addCommand(ReplaceCmd);
      f_mEDbD.setCommandListener(this);

      ByteArrayInputStream bais;
      DataInputStream dis;

      String inlname;   String infname;
      String inemail; String inphone;
      String inmobile; String infax;
      String instreet; String inzip;
      String incity;    String incountry;

      int recId;
      RecordEnumeration enum =
                  DbAdBok.enumerateRecords(null,null,false);
      recId = enum.nextRecordId();
      bais = new ByteArrayInputStream(DbAdBok.getRecord(recId));
      dis  = new DataInputStream(bais);
      DbAdBok.deleteRecord(recId);

      inlname  = dis.readUTF();
      lname = new TextField("Last Name:", inlname, 32,
                  TextField.ANY);

      infname = dis.readUTF();
```

```
      fname = new TextField("First Name:", infname, 32,
                  TextField.ANY);

      inemail = dis.readUTF();
      email = new TextField("Email:", inemail, 32,
                  TextField.EMAILADDR);

      inphone = dis.readUTF();
      phone = new TextField("Phone:", inphone, 32,
                  TextField.PHONENUMBER);

      inmobile = dis.readUTF();
      mobile = new TextField("Mobile:", inmobile, 32,
                   TextField.PHONENUMBER);

      infax = dis.readUTF();
      fax = new TextField("Fax:", infax, 32,
                TextField.PHONENUMBER);

      instreet = dis.readUTF();
      street = new TextField("Street:", instreet, 32,
                   TextField.ANY);

      inzip = dis.readUTF();
      zip = new TextField("Zip:", inzip, 32, TextField.NUMERIC);

      incity = dis.readUTF();
      city = new TextField("City:", incity, 32, TextField.ANY);

      incountry = dis.readUTF();
      country = new TextField("Country:", incountry, 32,
                    TextField.ANY);

      f_mEDbD.append(lname);        f_mEDbD.append(fname);
      f_mEDbD.append(email);        f_mEDbD.append(phone);
      f_mEDbD.append(mobile);       f_mEDbD.append(fax);
      f_mEDbD.append(street);       f_mEDbD.append(zip);
      f_mEDbD.append(city);         f_mEDbD.append(country);
      display.setCurrent(f_mEDbD);
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e)      {
}
catch(RecordStoreException e){
}
}

public void editDbData(){
try{
      ByteArrayOutputStream baos = new ByteArrayOutputStream();
      DataOutputStream dos = new DataOutputStream(baos);

      dos.writeUTF(lname.getString());
```

```
        byte[] blname = baos.toByteArray();
        DbAdBok.addRecord(blname, 0, blname.length);

        dos.writeUTF(fname.getString());
        byte[] bfname = baos.toByteArray();
        DbAdBok.addRecord(bfname, 0, bfname.length);


        dos.writeUTF(email.getString());
        byte[] bemail = baos.toByteArray();
        DbAdBok.addRecord(bemail, 0, bemail.length);

        dos.writeUTF(phone.getString());
        byte[] bphone = baos.toByteArray();
        DbAdBok.addRecord(bphone, 0, bphone.length);

        dos.writeUTF(mobile.getString());
        byte[] bmobile = baos.toByteArray();
        DbAdBok.addRecord(bmobile, 0, bmobile.length);

        dos.writeUTF(fax.getString());
        byte[] bfax = baos.toByteArray();
        DbAdBok.addRecord(bfax, 0, bfax.length);

        dos.writeUTF(street.getString());
        byte[] bstreet = baos.toByteArray();
        DbAdBok.addRecord(bstreet, 0, bstreet.length);

        dos.writeUTF(zip.getString());
        byte[] bzip = baos.toByteArray();
        DbAdBok.addRecord(bzip, 0, bzip.length);

        dos.writeUTF(city.getString());
        byte[] bcity = baos.toByteArray();
        DbAdBok.addRecord(bcity, 0, bcity.length);

        dos.writeUTF(country.getString());
        byte[] bcountry = baos.toByteArray();
        DbAdBok.addRecord(bcountry, 0, bcountry.length);

        mainMenu();
    }
    catch(IOException e){
    }
    catch(RecordStoreFullException e){
    }
    catch(RecordStoreNotOpenException e){
    }
    catch(RecordStoreException e){
    }
    }

    public void mulaiHapusDb(){
        if(RecordStore.listRecordStores() == null ){
```

```
            f = new Form("Erase");
            f.append("(Empty)");
            f.addCommand(backCommand);
            f.setCommandListener(this);
            display.setCurrent(f);
      }
      if(RecordStore.listRecordStores() != null ){
            f =   new Form("Erase");
            dt =  new ChoiceGroup("Select Entry:",Choice.EXCLUSIVE);
            String allDbs[] = RecordStore.listRecordStores();
            for(int i = 0; i < allDbs.length; i++){
                  dt.append(allDbs[i],null);
            }
            f.append(dt);
            f.addCommand(EraseCmd);
            f.addCommand(backCommand);
            f.setCommandListener(this);
            display.setCurrent(f);
      }
}

public void konfHapusDb(){
      f = new Form("Address Book");
      String dbName = dt.getString(dt.getSelectedIndex());
      f.append("Delete Entry:" + dbName+ "?");
      f.addCommand(YesCmd);
      f.addCommand(backCommand);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void hapusDb(){
try {
      display = Display.getDisplay(this);
      a = new Alert("Alert");
      a.setType(AlertType.WARNING);
      String allDbs[] = RecordStore.listRecordStores();
      for(int i = 0; i < allDbs.length; i++){
            dt.append(allDbs[i],null);
      }
      String dbName = dt.getString(dt.getSelectedIndex());
      dt.delete(dt.getSelectedIndex());
      RecordStore.deleteRecordStore(dbName);
      a.setString("Entry " + dbName+ "Erased !");
      display.setCurrent(a);
}
catch(RecordStoreNotFoundException e){
}
catch(RecordStoreException e){
}
}

public void tutupDbed(){
try{
```

```
      DbAdBok.closeRecordStore();
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
catch(NullPointerException e){
      mulaiEditDb();
}
}

public void tutupDbvi(){
try{
      DbAdBok.closeRecordStore();
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
catch(NullPointerException e){
      mulaiTampilDb();
}
}



public void tutupDber(){
try{
      DbAdBok.closeRecordStore();
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
catch(NullPointerException e){
      mulaiHapusDb();
}
}

public void testadd() {
      mulaiTambahDb();
      currentMenu = "Add";
}

public void testview() {
      tutupDbvi();
      mulaiTampilDb();
      currentMenu = "View";
}

public void testedit() {
      tutupDbed();
      mulaiEditDb();
      currentMenu = "Edit";
```

```
}

public void testerase() {
      tutupDber();
      mulaiHapusDb();
      currentMenu = "Erase";
}

public void commandAction(Command c, Displayable d) {
      String label = c.getLabel();
      if (label.equals("Exit")) {
            destroyApp(true);
      }
      if(label.equals("Add")){
            tambahDb();
      }
      if(label.equals("Save")){
            tambahDbData();
      }
      if(label.equals("Detail")){
            bukaTampilDb();
      }
      if(label.equals("Edit")){
            bukaEditDb();
      }

      if(label.equals("Replace")){
            editDbData();
      }
      if(label.equals("Erase")){
            konfHapusDb();
      }
      if(label.equals("Yes")){
            hapusDb();
            mainMenu();
      }
      else if (label.equals("Back")) {
            if(currentMenu.equals("Add") ||
            currentMenu.equals("View") ||
            currentMenu.equals("Edit") ||
            currentMenu.equals("Erase")) {
                        mainMenu();
            }
            }
      else {
            List down = (List)display.getCurrent();
            switch(down.getSelectedIndex()) {
                  case 0: testadd();break;
                  case 1: testview();break;
                  case 2: testedit();break;
                  case 3: testerase();break;
            }
            }
      }
```

```
}
```

12. Listing Program "Scheduler"

```
import java.io.*;
import java.util.*;
import javax.microedition.rms.*;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;

public class sched extends MIDlet implements CommandListener{

Display display = null;
List menu = null;
TextBox input = null;

static final Command backCmd = new Command("Back", Command.BACK, 0);
static final Command mainMenuCmd = new Command("Main",
Command.SCREEN, 1);
static final Command exitCmd = new Command("Exit", Command.STOP, 2);
private Command saveCmd = new Command("Save", Command.OK, 1);
private Command eraseCmd = new Command("Erase", Command.OK, 1);
private Command editCmd = new Command("Edit",Command.OK,1);
private Command replaceCmd = new Command("Replace",Command.OK,1);
private Command addCmd = new Command("Add", Command.OK, 1);
private Command yesCmd = new Command("Yes", Command.OK, 1);
private Command stopCmd = new Command("Stop", Command.STOP, 1);
String currentMenu = null;

private Form f, f2;
private DateField df, dfc;
private TextField t;
private Timer timer;
private TimerTask task;
private ChoiceGroup cg, dt;
private Date date, time, setDate, setCurDate, curDate;
private Calendar c, cd, ccd, cur;
private Alert a, alarm;
private StringItem si;
private RecordStore DbPesan, DbWaktu;

public sched(){
}

public void startApp() throws MIDletStateChangeException{
display = Display.getDisplay(this);
a = new Alert("Scheduler");
alarm = new Alert("Scheduler");
alarm.setType(AlertType.WARNING);
alarm.setTimeout(20000);
menu = new List("Scheduler", Choice.IMPLICIT);
menu.append("Date/Time", null);
```

```
menu.append("Make Notes", null);
menu.append("View Notes", null);
menu.append("Edit Notes", null);
menu.append("Erase Notes", null);
menu.append("Reset Notes", null);
menu.addCommand(exitCmd);
menu.setCommandListener(this);
mainMenu();
}
public void pauseApp(){
      display = null;
      menu = null;
      input = null;
}

public void destroyApp(boolean unconditional) {
notifyDestroyed();
}

void mainMenu(){
      display.setCurrent(menu);
      currentMenu = "Main";
}

public void mulai(){
if(RecordStore.listRecordStores() == null ){
      try{
            DbPesan = RecordStore.openRecordStore("pesan",true);
            DbWaktu = RecordStore.openRecordStore("waktu",true);
            DbPesan.closeRecordStore();
            DbWaktu.closeRecordStore();
            mulaiMakePesan();
}
catch(RecordStoreException e){
}
}
if(RecordStore.listRecordStores() != null ){
      mulaiMakePesan();
}
}

public void currentDateTime(){
      f = new Form("Date/Time");
      dfc = new DateField("", DateField.DATE_TIME);
      dfc.setDate(new Date());
      f.append(dfc);
      f.setCommandListener(this);
      f.addCommand(backCmd);
      display.setCurrent(f);
}

public void mulaiMakePesan(){
      f = new Form("Make Notes");
      t = new TextField("Subject:", null, 32, TextField.ANY);
```

```
        f.append(t);
        f.setCommandListener(this);
        f.addCommand(backCmd);
        f.addCommand(addCmd);
        display.setCurrent(f);
}

public void mulaiMakeWaktu(){
        f = new Form("Set Date/Time");
        df = new DateField("", DateField.DATE_TIME);
        f.append(df);
        f.setCommandListener(this);
        f.addCommand(backCmd);
        f.addCommand(saveCmd);
        display.setCurrent(f);
}

public static Date setWaktu(Date date){
Calendar cd = Calendar.getInstance();
cd.setTime( date );
cd.set( Calendar.MONTH, cd.get( Calendar.MONTH ) );
cd.set( Calendar.DAY_OF_MONTH, cd.get( Calendar.DAY_OF_MONTH ) );
cd.set( Calendar.YEAR, cd.get( Calendar.YEAR ) );
cd.set( Calendar.HOUR_OF_DAY, cd.get( Calendar.HOUR_OF_DAY ) );
cd.set( Calendar.MINUTE, cd.get( Calendar.MINUTE ) );
cd.set( Calendar.SECOND, 0);
cd.set( Calendar.MILLISECOND, 0);
return cd.getTime();
}

public static Date setCurWaktu(Date curDate){
Calendar ccd = Calendar.getInstance();
ccd.setTime( curDate );
ccd.set( Calendar.MONTH, ccd.get( Calendar.MONTH ) );
ccd.set( Calendar.DAY_OF_MONTH, ccd.get( Calendar.DAY_OF_MONTH ) );
ccd.set( Calendar.YEAR, ccd.get( Calendar.YEAR ) );
ccd.set( Calendar.HOUR_OF_DAY, ccd.get( Calendar.HOUR_OF_DAY ) );
ccd.set( Calendar.MINUTE, ccd.get( Calendar.MINUTE ) );
ccd.set( Calendar.SECOND, 0);
ccd.set( Calendar.MILLISECOND, 0);
return ccd.getTime();
}

public void proses(){
Date date = df.getDate();
        setDate = setWaktu(date);
        Calendar c = Calendar.getInstance();
        c.setTime(setDate);
        Timer timer = new Timer();
        TimerTask task = new myTask();
        timer.schedule(task, c.getTime());
        bukaTambahDbWaktu();
}
```

```
public class myTask extends TimerTask{
      public void run(){
      Date curDate = new Date();
      setCurDate = setCurWaktu(curDate);
      Calendar cur = Calendar.getInstance();
      cur.setTime(setCurDate);
      bukaBandingPesanWaktu();
}
}

public void bukaTambahDbPesan(){
try{
      DbPesan = RecordStore.openRecordStore("pesan",false);
      tambahDbPesan();
}
catch(RecordStoreException e){
}
}

public void tambahDbPesan(){
try{
ByteArrayOutputStream baosPesan = new ByteArrayOutputStream();
DataOutputStream dosPesan = new DataOutputStream(baosPesan);
dosPesan.writeUTF(t.getString());
byte[] bPesan = baosPesan.toByteArray();
DbPesan.addRecord(bPesan, 0, bPesan.length);
mulaiMakeWaktu();
}

catch(IOException e){
}
catch(RecordStoreFullException e){
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
}

public void bukaTambahDbWaktu(){
try{
DbWaktu = RecordStore.openRecordStore("waktu",false);
tambahDbWaktu();
}
catch(RecordStoreException e){
}
}

public void tambahDbWaktu(){
try{
ByteArrayOutputStream baosWaktu = new ByteArrayOutputStream();
DataOutputStream dosWaktu = new DataOutputStream(baosWaktu);
dosWaktu.writeUTF(setDate.toString());
byte[] bWaktu = baosWaktu.toByteArray();
```

```
DbWaktu.addRecord(bWaktu, 0, bWaktu.length);
mainMenu();
}
catch(IOException e){
}
catch(RecordStoreFullException e){
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
}

public void bukaTampilDbPesan(){
try{
DbPesan = RecordStore.openRecordStore("pesan",false);
tampilDbPesan();
}
catch(RecordStoreException e){
}
}

public void tampilDbPesan(){
try{
f = new Form("View Notes");
ChoiceGroup cg = new ChoiceGroup("",Choice.EXCLUSIVE);
f.addCommand(backCmd);
f.setCommandListener(this);
f.append(cg);
ByteArrayInputStream baisPesan;
DataInputStream disPesan;
String inPesan;
RecordEnumeration enum = DbPesan.enumerateRecords(null,null, false);
while(enum.hasNextElement()){
int recId   = enum.nextRecordId();
baisPesan = new ByteArrayInputStream(DbPesan.getRecord(recId));
disPesan  = new DataInputStream(baisPesan);
      inPesan   = disPesan.readUTF();
      cg.append(inPesan,null);
}
display.setCurrent(f);
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
}

public void bukaBandingPesanWaktu(){
try{
DbWaktu = RecordStore.openRecordStore("waktu",false);
DbPesan = RecordStore.openRecordStore("pesan",false);
```

```
bandingPesanWaktu();
}
catch(RecordStoreException e){
}
}

public void bandingPesanWaktu(){
try{
ByteArrayInputStream baisPesan;
DataInputStream disPesan;
String inPesan;
ByteArrayInputStream baisWaktu;
DataInputStream disWaktu;
String inWaktu;
RecordEnumeration enum = DbWaktu.enumerateRecords(null,null, false);
while(enum.hasNextElement()){
int recId = enum.nextRecordId();
baisWaktu = new ByteArrayInputStream(DbWaktu.getRecord(recId));
disWaktu  = new DataInputStream(baisWaktu);
inWaktu   = disWaktu.readUTF();
String strInWaktu = inWaktu.toString();
String strSetCurDate = setCurDate.toString();
int i = strInWaktu.compareTo(strSetCurDate);

baisPesan = new ByteArrayInputStream(DbPesan.getRecord(recId));
disPesan  = new DataInputStream(baisPesan);
inPesan   = disPesan.readUTF();
String strNoPesan = "";
String strInPesan = inPesan;
int j = strNoPesan.compareTo(strInPesan);
if(i == 0 && j !=0 ){
f = new Form("");
f.addCommand(stopCmd);
f.setCommandListener(this);
alarm.setString(inPesan);
display.setCurrent(alarm, f);
break;
}
}
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
}

public void bukaEditDb(){
try{
DbPesan = RecordStore.openRecordStore("pesan",false);
tampilEditDb();
}
catch(RecordStoreNotFoundException e){
```

```
}
catch(RecordStoreException e){
}
}

public void tampilEditDb(){
try{
f2   = new Form("Edit Notes");
dt = new ChoiceGroup("",Choice.EXCLUSIVE);
f2.addCommand(backCmd);
f2.addCommand(editCmd);
f2.setCommandListener(this);
f2.append(dt);
ByteArrayInputStream bais;
DataInputStream dis;
String in;
RecordEnumeration enum = DbPesan.enumerateRecords(null,null, false);
while(enum.hasNextElement()){
int recId = enum.nextRecordId();
bais = new ByteArrayInputStream(DbPesan.getRecord(recId));
dis  = new DataInputStream(bais);
in   = dis.readUTF();
dt.append(in,null);
}
display.setCurrent(f2);
}
catch(IOException e){
}

catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
}

public void mulaiEditDb(){
      f = new Form("Edit Notes");
      t = new TextField("Subject:", null, 32, TextField.ANY);
      f.append(t);
      f.setCommandListener(this);
      f.addCommand(backCmd);
      f.addCommand(replaceCmd);
      display.setCurrent(f);
}

public void replaceDb(){
try{
String strData = dt.getString(dt.getSelectedIndex());
ByteArrayInputStream bais;
DataInputStream dis;
String in;
RecordEnumeration enum = DbPesan.enumerateRecords(null,null, false);
while(enum.hasNextElement()){
      int recId = enum.nextRecordId();
```

```
      bais = new ByteArrayInputStream(DbPesan.getRecord(recId));
      dis  = new DataInputStream(bais);
      in   = dis.readUTF();
      if (in.equals(strData)){
            DbPesan.deleteRecord(recId);
            break;
      }
}
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e){
}
catch(RecordStoreException e){
}
bukaTambahDbPesan();
}

public void bukaHapusDb(){
try{
DbPesan = RecordStore.openRecordStore("pesan",false);
tampilHapusDb();
}
catch(RecordStoreNotFoundException e){
}
catch(RecordStoreException e){
}
}

public void tampilHapusDb(){
try{
f2   = new Form("Erase Notes");
dt = new ChoiceGroup("",Choice.EXCLUSIVE);
f2.addCommand(backCmd);
f2.addCommand(eraseCmd);
f2.setCommandListener(this);
f2.append(dt);
ByteArrayInputStream bais;
DataInputStream dis;
String in;
RecordEnumeration enum = DbPesan.enumerateRecords(null,null, false);
while(enum.hasNextElement()){
int recId = enum.nextRecordId();
bais = new ByteArrayInputStream(DbPesan.getRecord(recId));
dis  = new DataInputStream(bais);
in   = dis.readUTF();
dt.append(in,null);
}
display.setCurrent(f2);
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e){
}
```

```
catch(RecordStoreException e){
}
}

public void hapusDataDb(){
try{
String strData = dt.getString(dt.getSelectedIndex());
ByteArrayInputStream bais;
DataInputStream dis;
String in;
RecordEnumeration enum = DbPesan.enumerateRecords(null,null, false);
while(enum.hasNextElement()){
      int recId   = enum.nextRecordId();
      bais = new ByteArrayInputStream(DbPesan.getRecord(recId));
      dis  = new DataInputStream(bais);
      in   = dis.readUTF();
      if (in.equals(strData)){
            DbPesan.deleteRecord(recId);
            break;
      }
}
mainMenu();
}
catch(IOException e){
}
catch(RecordStoreNotOpenException e){
}


catch(RecordStoreException e){
}
}

public void mulaiResetDb(){
f = new Form("Reset Notes");
      f.append("Reset All Notes ?");
      f.addCommand(backCmd);
      f.addCommand(yesCmd);
      f.setCommandListener(this);
      display.setCurrent(f);
}

public void tutupDbvi(){
            try{
                  DbPesan.closeRecordStore();
                  DbWaktu.closeRecordStore();
            }
            catch(RecordStoreNotOpenException e){
            }
            catch(RecordStoreException e){
            }
            catch(NullPointerException e){
                  bukaTampilDbPesan();
            }
```

```
}

public void tutupDbed(){
            try{
                  DbPesan.closeRecordStore();
                  DbWaktu.closeRecordStore();
            }
            catch(RecordStoreNotOpenException e){
            }
            catch(RecordStoreException e){
            }
            catch(NullPointerException e){
                  bukaEditDb();
            }
      }
      public void tutupDbre(){
            try{
                  DbPesan.closeRecordStore();
                  DbWaktu.closeRecordStore();
            }
            catch(RecordStoreNotOpenException e){
            }
            catch(RecordStoreException e){
            }
            catch(NullPointerException e){
                  mulaiResetDb();
            }
      }

public void resetDb(){
try{
RecordStore.deleteRecordStore("pesan");
RecordStore.deleteRecordStore("waktu");
}
catch(RecordStoreNotFoundException e){
}
catch(RecordStoreException e){
}
}

public void testdate(){
      currentDateTime();
      currentMenu = "date";
}

public void testmake(){
      mulai();
      currentMenu = "make";
}

public void testview(){
      tutupDbvi();
      bukaTampilDbPesan();
      currentMenu = "view";
```

```
}

public void testedit(){
      tutupDbed();
      bukaEditDb();
      currentMenu = "edit";
}

public void testerase(){
      bukaHapusDb();
      currentMenu = "erase";
}

public void testreset(){
      tutupDbre();
      mulaiResetDb();
      currentMenu = "reset";
}
public void commandAction(Command c, Displayable d){
      String label = c.getLabel();
      if (label.equals("Exit")){
                  destroyApp(true);
      }
      if (label.equals("Save")){
            proses();
      }
      if (label.equals("Add")){
            bukaTambahDbPesan();
      }
      if (label.equals("Erase")){
            hapusDataDb();
      }
      if (label.equals("Edit")){
            mulaiEditDb();
      }
      if (label.equals("Replace")){
            replaceDb();
      }
      if (label.equals("Yes")){
            resetDb();
            mainMenu();
      }
      if (label.equals("Stop")){
            destroyApp(true);
      }
      else if (label.equals("Back")){
            if(currentMenu.equals("date") ||
            currentMenu.equals("make") ||
            currentMenu.equals("view") ||
            currentMenu.equals("edit") ||
            currentMenu.equals("erase")||
            currentMenu.equals("reset") ){
            mainMenu();
}
```

```
        }
        else{
              List down = (List)display.getCurrent();
              switch(down.getSelectedIndex()){
                    case 0: testdate();break;
                    case 1: testmake();break;
                    case 2: testview();break;
                    case 3: testedit();break;
                    case 4: testerase();break;
                    case 5: testreset();break;
              }
        }
}
}
```