package hello;

import java.util.Date;

import java.util.Timer;

import java.util.TimerTask;

import javax.microedition.lcdui.Alert;

import javax.microedition.lcdui.AlertType;

import javax.microedition.lcdui.Command;

import javax.microedition.lcdui.CommandListener;

import javax.microedition.lcdui.DateField;

import javax.microedition.lcdui.Display;

import javax.microedition.lcdui.Displayable;

import javax.microedition.lcdui.Form;

import javax.microedition.lcdui.Item;

import javax.microedition.lcdui.ItemStateListener;

import javax.microedition.midlet.MIDlet;

public class HelloMIDlet extends MIDlet implements ItemStateListener, CommandListener {

private Command alarmCommand= new Command("alarm", Command.SCREEN, 1);

private Command resetCommand= new Command("Reset", Command.SCREEN, 1);

private Command exitCommand= new Command("Exit", Command.EXIT, 1);

private Display display;

private DateField alarmTimeSetting;

private Form aForm= new Form("Alarm Clock");

private int dateIndex;

private Date currentTime= new Date();

private Timer aTimer;

private boolean dateOK = false;

public HelloMIDlet() {

display = Display.getDisplay(this);

alarmTimeSetting = new DateField("", DateField.DATE\_TIME);

alarmTimeSetting.setDate(currentTime);

dateIndex = aForm.append(alarmTimeSetting);

aForm.addCommand(alarmCommand);

aForm.addCommand(resetCommand);

aForm.addCommand(exitCommand);

aForm.setCommandListener(this);

aForm.setItemStateListener(this);

}

public void startApp() {

display.setCurrent(aForm);

}

public void pauseApp() {

}

public void destroyApp(boolean unconditional) {

}

public void itemStateChanged(Item item) {

if (item == alarmTimeSetting) {

if (alarmTimeSetting.getDate().getTime() < currentTime.getTime())

dateOK = false;

else

dateOK = true;

}

}

public void commandAction(Command c, Displayable s) {

if (c == alarmCommand) {

if (dateOK == false) {

Alert aAlert = new Alert("Cannot set", "Warning", null, null);

aAlert.setTimeout(Alert.FOREVER);

aAlert.setType(AlertType.ERROR);

display.setCurrent(aAlert);

} else {

aTimer = new Timer();

alarmTimerTask aTimerTask = new alarmTimerTask();

long amount = alarmTimeSetting.getDate().getTime() - currentTime.getTime();

aTimer.schedule(aTimerTask, amount);

aForm.removeCommand(alarmCommand);

aForm.removeCommand(resetCommand);

aForm.delete(dateIndex);

aForm.setTitle("Waiting....");

}

} else if (c == resetCommand) {

alarmTimeSetting.setDate(currentTime = new Date());

} else if (c == exitCommand) {

destroyApp(false);

notifyDestroyed();

}

}

class alarmTimerTask extends TimerTask {

public final void run() {

Alert aAlert = new Alert("Alert!");

aAlert.setTimeout(Alert.FOREVER);

aAlert.setType(AlertType.ALARM);

AlertType.ERROR.playSound(display);

display.setCurrent(aAlert);

cancel();

}

}

}