Importing calendar event in Java ME

This code example demonstrates how to import a calendar event from a file.

Overview

To import a calendar event from a file:

1. Open the file which contains the event data and open the input stream from this file.
2. Set the input stream as a parameter in `PIM.fromSerialFormat`. This method returns an array of imported PIM items. The returned array may include also other content than calendar events so additional checking is needed.
3. Make a copy of each imported event by calling the `EventList.importEvent` method.
4. Save each copy in the events list by executing the `Event.commit` method for each copied event object.
5. Close the file.

This MIDlet imports events from files in "UTF-8" character encoding format without any transfer character encoding. Event data in the file can be in vCalendar format.

Source file: ImportCalendarEvent.java

```
import javax.microedition.midlet.MIDlet;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.List;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.Alert;

import javax.microedition.pim.PIM;
import javax.microedition.pim.PIMItem;
import javax.microedition.pim.EventList;
import javax.microedition.pim.Event;
import javax.microedition.pim.PIMException;

import javax.microedition.io.Connector;
import javax.microedition.io.file.FileConnection;

import java.util.Enumeration;
import java.io.InputStream;


public class ImportCalendarEvent extends MIDlet implements CommandListener  {

    private Display display;

    // List where events will be placed.
    private List eventListCtrl;

    // Command for exporting selected event. Placed on eventListCtrl.
    private Command cmdImportEvent;
    // Command for exiting from application. Placed on eventListCtrl.
    private Command cmdExit;

    // Name of file with event data to be imported.
    private final String IMPORT_FILENAME = "importEvent.txt";

    /**
     * Constructor
     */
    public ImportCalendarEvent() {
        if(checkPIMSupport() == false) {
            exitMIDlet();
        }

        initializeComponents();
    }

    /**
     * Initializes components of midlet.
     */
    private void initializeComponents() {
        // Get display
        display = Display.getDisplay(this);

        // Create list control
        eventListCtrl = new List("Events", List.IMPLICIT);

        cmdImportEvent = new Command("Import event", Command.SCREEN, 0);
        eventListCtrl.addCommand(cmdImportEvent);

        cmdExit = new Command("Exit", Command.EXIT, 0);
        eventListCtrl.addCommand(cmdExit);

        eventListCtrl.setCommandListener(this);

        // Add events to list control.
        addEventsToListCtrl();
    }

    /**
     * Checks PIM support.
     * @return - true if PIM is supported, false otherwise.
     */
    private boolean checkPIMSupport() {
        String propValue = System.getProperty("microedition.pim.version");
        if(propValue != null) {
            return true;
        } else {
```


(C) Copyright Nokia 2013. All rights reserved.

```
        return false;
    }
}

/**
 * Adds events to list.
 */
private void addEventsToListCtrl() {
    try {
        // Get list of events.
        EventList eventList = (EventList)PIM.getInstance().openPIMList(
                PIM.EVENT_LIST, PIM.READ_WRITE);

        // Check if "summary" and "uid" fields of event item are supported.
        // If not supported, exit from application.
        if(eventList.isSupportedField(Event.SUMMARY) == false) {
            throw new Exception();
        }

        // Clear list control.
        eventListCtrl.deleteAll();

        // Add events to list control.
        Enumeration events = eventList.items();
        while(events.hasMoreElements() == true) {
            Event event = (Event)events.nextElement();
            String eventSummary = event.getString(Event.SUMMARY,
                    PIMItem.ATTR_NONE);
            // Add event's summary to list control
            eventListCtrl.append(eventSummary, null);
        }

        // Close list of events.
        eventList.close();

    } catch(PIMException pimExc) {
        // TODO: Handle error on working with PIM.
    } catch(SecurityException secExc) {
        // TODO: Handle error on access to PIM.
    } catch(Exception exc) {
        // If unknown error was catched, exit from application.
        exitMIDlet();
    }
}

/**
 * Imports event from file placed private folder.
 * @return true if event was imported successfully, false - otherwise
 */
private boolean importEventFromFile() {
    try {
        // Open file for importing event.
        String path = System.getProperty("fileconn.dir.memorycard");
        String fileName = path + IMPORT_FILENAME;

        FileConnection file = (FileConnection)Connector.open(fileName,
                Connector.READ_WRITE);

        // If there is no file then create it
        if(file.exists() == false) {
            throw new Exception("File for importing was not found.");
        }

        // Export event.
        InputStream inStream = file.openInputStream();

        PIM pimInst = PIM.getInstance();
        PIMItem[] items = pimInst.fromSerialFormat(inStream, "UTF-8");
        // Assume what there is only one event item in file.
        if(items.length > 0) {
            importEventToList((Event)items[0]);
        }

        // Close file.
        inStream.close();
        file.close();

        showAlert("Info", "Event was imported from file.");

    } catch(Exception exc) {
        showAlert("Error", exc.getMessage());
        return false;
    }

    return true;
}

/**
 * Imports specified event to the list of events.
 * @param importedEvent - event to be imported
 * @throws javax.microedition.pim.PIMException on PIM error
 */
private void importEventToList(Event importedEvent) throws PIMException {
    // Open list of events
    EventList eventList = (EventList)PIM.getInstance().openPIMList(
            PIM.EVENT_LIST, PIM.READ_WRITE);

    // Event list creates new event based on imported event
    Event newEvent = eventList.importEvent(importedEvent);

    // Save new event to list
    newEvent.commit();
    // Close list of events
    eventList.close();
}

/**
 * Shows alert with specified title and text.
 * @param title - Title of alert.
 * @param message - text of alert.
 */
private void showAlert(String title, String message) {
    Alert alert = new Alert(title);
    alert.setString(message);
    alert.setTimeout(Alert.FOREVER);
    display.setCurrent(alert);
}

/**
 * From MIDlet.
```


(C) Copyright Nokia 2013. All rights reserved.

```
     * Signals the MIDlet that it has entered the Active state.
     */
    public void startApp() {
        display.setCurrent(eventListCtrl);
    }

    /**
     * From MIDlet.
     * Signals the MIDlet to enter the Paused state.
     */
    public void pauseApp() {
        // No implementation required.
    }

    /**
     * From MIDlet.
     * Signals the MIDlet to terminate and enter the Destroyed state.
     */
    public void destroyApp(boolean unconditional) {
        // No implementation required.
    }

    /**
     * Performs exit from midlet.
     */
    private void exitMIDlet() {
        notifyDestroyed();
    }

    /**
     * From CommandListener.
     * Indicates that a command event has occurred on Displayable displayable.
     * @param command - a Command object identifying the command.
     * @param displayable - the Displayable on which this event has occurred.
     */
    public void commandAction(Command command, Displayable displayable) {
        // Handles "show details" command on selected event.
        if(command == cmdImportEvent) {
            if(importEventFromFile() == true) {
                addEventsToListCtrl();
            }
        }
        // Handles "exit" command.
        if(command == cmdExit) {
            exitMIDlet();
        }
    }
}
```

Postconditions

This code snippet represents a MIDlet with a list control which contains summaries of events from the PIM calendar events list.

By pressing the "Import event" command the user can import an event from a file. The file containing the event that will be imported should be placed in the root directory of the memory card and named as an IMPORT_FILENAME constant. After the event has been imported, the list control with summaries of events will show all events in the list of events.

Supplementary material

Executables and source files are available in Media:ImportingCalendarEvent.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.