Exporting a calendar event in Java ME

Overview

This code snippet demonstrates how to export calendar events to a file in a specified format.

To export a calendar event to a file:

1. Get a list of available formats for exporting event records by calling `PIM.supportedSerialFormats`. In this snippet, the first avaliable format from the list is used.
2. Open the file and create an output stream to it.
3. Give the output file stream, the event that should be exported, encoding and format name as parameters to the `PIM.toSerialFormat` method. This method exports the event to a file using a specified format and encoding.
4. Close the file.

Source file: ExportCalendarEvent.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.util.Vector;
import java.io.OutputStream;

public class ExportCalendarEvent 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 cmdExportEvent;
    // Command for exiting from application. Placed on eventListCtrl.
    private Command cmdExit;

    // Contains events retrieved from PIM event list.
    private Vector eventsArray;

    private final String EXPORT_FILENAME = "exportEvent.txt";

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

        initializeComponents();
    }

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

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

        cmdExportEvent = new Command("Export event", Command.SCREEN, 0);
        eventListCtrl.addCommand(cmdExportEvent);

        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) {
```


(C) Copyright Nokia 2013. All rights reserved.

```
            return true;
        } else {
            return false;
        }
    }

    /**
     * Adds events to list.
     */
    private void addEventsToListCtrl() {
        try {
            // Create hashtable.
            eventsArray = new Vector();

            // 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 ||
               eventList.isSupportedField(Event.UID) == false) {
                throw new Exception();
            }

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

    /**
     * Exports selected event to file in private folder.
     */
    private void exportEvent() {
        try {
            // Get supported formats.
            PIM pimInst = PIM.getInstance();
            String[] formats = pimInst.supportedSerialFormats(PIM.EVENT_LIST);
            if(formats.length == 0) {
                throw new Exception("No formats for event exporting were found.");
            }

            // Get event
            int eventIndex = eventListCtrl.getSelectedIndex();
            Event event = (Event)eventsArray.elementAt(eventIndex);
            if(event == null) {
                throw new Exception("No events for exporting were found.");
            }

            // Open file for exporting event.
            String path = System.getProperty("fileconn.dir.memorycard");
            String fileName = path + EXPORT_FILENAME;

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

            // If there is no file then create it
            if(file.exists() == false) {
                file.create();
            }

            // Export event.
            OutputStream outStream = file.openOutputStream();
            pimInst.toSerialFormat(event, outStream, "UTF-8", formats[0]);

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

            showAlert("Info", "Event has been exported to " + fileName);

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

    /**
     * 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.
     * 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() {
```


(C) Copyright Nokia 2013. All rights reserved.

```
        // 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 == cmdExportEvent) {
            exportEvent();
        }
        // Handles "exit" command.
        if(command == cmdExit) {
            exitMIDlet();
        }
    }
}
```

Postconditions

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

By choosing an event from the list and pressing the "Export event" menu command the user can export the selected event to file.

The file with the exported event will be created in the root directory of the memory card and have the name EXPORT_FILENAME constant. The event format will be chosen from the list of provided formats. The exported file will have UTF-8 character encoding.

Supplementary material

Executables and source files are available at Media:ExportingCalendarEvent.zip.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.