Deleting calendar event in Java ME

Overview

This code snippet demonstrates how to delete an event from the calendar using the PIM API.

The list of events can be retrieved using `PIM.openPIMList`. For more information, refer to the code snippet Listing calendar events in Java ME.

After the event that should be deleted is found, and we have an `Event` reference, the method `EventList.removeEvent` is called to delete the event.

Preconditions

In order to test this snippet there should be at least one event in the database. The application also needs permission to read and write PIM database data.

Source

```
private List listEvents;

/**
 * Holds reference to the currently opened event list.
 */
private EventList eventList;

private static final Command DELETE_COMMAND =
        new Command("Delete", Command.ITEM, 0);
```

When an event is selected in the list, delete it:

```
listEvents.setSelectCommand(DELETE_COMMAND);
listEvents.setCommandListener(this);
```

```
/**
 * Deletes an event by it's index in the event list.
 * @param itemIndex index of the event in the event list.
 */
private void deleteEvent(int itemIndex) {
    int i = 0;
    Enumeration eventItems = null;
    Event event = null;
    try {
        printString("Getting event list items...");
        eventItems = eventList.items();
    } catch (PIMException ex) {
        releaseEventList();
        printString(ex.toString());
        return;
    }
    // Checking event count
    if (!eventItems.hasMoreElements()) {
        printString("Event list is empty!");
    }
    event = (Event)eventItems.nextElement();
    while ((i != itemIndex) && (eventItems.hasMoreElements())) {
        event = (Event)eventItems.nextElement();
        i++;
    }
    if (event == null) {
        return;
    }
    try {
        printString("Removing event...");
        eventList.removeEvent(event);
    } catch (PIMException ex) {
        printString(ex.toString());
    }
}
```

```
/**
 * From CommandListener.
 * Called by the system to indicate that a command has been invoked on a
 * particular displayable.
 * @param command the command that was invoked
 * @param displayable the displayable where the command was invoked
 */
public void commandAction(Command command, Displayable displayable) {
    if (command == DELETE_COMMAND) {
        // Get the currently selected event
        int i = listEvents.getSelectedIndex();
        String item = listEvents.getString(i);
        printString("Selected event: " + item);
        // Bring the main form back to the foreground
        display.setCurrent(mainForm);
        deleteEvent(i);
        // Close the event list
        releaseEventList();
    }
}
```


(C) Copyright Nokia 2013. All rights reserved.

Postconditions

The selected event is removed from the calendar.

See also

- Listing calendar events in Java ME

Supplementary material

This code snippet is part of the stub concept, which means that it has been patched on top of a template application in order to be more useful to developers. The version of the Java ME stub application used as a template in this snippet is v1.1.

- The patched, executable application that can be used to test the features described in this snippet is available for download at Media:DeleteEvent.zip.
- You can view all the changes that are required to implement the above-mentioned features. The changes are provided in unified diff and colour-coded diff (HTML) formats in Media:DeleteEvent.diff.zip.
- For general information on applying the patch, see Using Diffs.
- For unpatched stub applications, see Example app stubs with logging framework.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.