Deleting a contact in Java ME

Overview

The following code snippet demonstrates how to delete a contact using the PIM API.

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

After the contact to be deleted is found, and we have a `Contact` reference, the `ContactList.removeContact` method is called to delete it.

Preconditions

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

Source

```
private List listContacts;

/**
 * Holds reference to the currently opened contact list.
 */
private ContactList contactList;

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

When a contact is selected in the list, delete it:

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

```
/**
 * Deletes a contact by it's index in the contact list.
 * @param itemIndex index of Contact in the contact list.
 */
private void deleteContact(int itemIndex) {
    int i = 0;
    Enumeration contacts = null;
    Contact contact = null;
    try {
        printString("Getting contact list items");
        contacts = contactList.items();
    } catch (PIMException ex) {
        releaseContactList();
        printString(ex.toString());
        return;
    }
    // Checking contact count
    if (!contacts.hasMoreElements()) {
        printString("Contact list is empty!");
    }
    contact = (Contact)contacts.nextElement();
    while ((i != itemIndex) && (contacts.hasMoreElements())) {
        contact = (Contact)contacts.nextElement();
        i++;
    }
    if (contact == null) {
        return;
    }
    try {
        printString("Removing contact ...");
        contactList.removeContact(contact);
    } 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 contact
        int i = listContacts.getSelectedIndex();
        String item = listContacts.getString(i);
        printString("Selected contact: " + item);
        // Bring the main form back to the foreground
        display.setCurrent(mainForm);
        deleteContact(i);
        // Close ContactList object instance
        releaseContactList();
    }
}
```


(C) Copyright Nokia 2013. All rights reserved.

Postconditions

The selected contact is removed from the phonebook.

See also

- Listing contacts 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 for 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:DeleteContact.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:DeleteContact.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.