How to use Commands in Java ME

This article explains the mapping rules between Commands and softkeys and Options menus for Nokia S60 and Series 40 devices.

Overview

```
Command(String shortLabel, String longLabel, int commandType, int priority)
```

In Nokia devices Commands have mapping rules to softkeys and Options menus that are based on Nokia user interface guidelines. It is essential to use logically proper and correct types for Commands. If right command types for the Commands intent are used then the mapping should work as expected and commands should be presented automatically according to Nokia Series 40 and S60 user interface guidelines (e.g. according to S60 UI style).

Commands have following types as specified in MIDP LCDUI API:

- OK
- ITEM
- SCREEN
- BACK
- CANCEL
- EXIT
- STOP
- HELP

These commands can be grouped to positive (OK, ITEM, SCREEN, HELP) and negative (BACK, CANCEL, EXIT, STOP).

Nokia user interface guidelines and style positions negative operations like backward navigation to right softkey. Positive operations like forward navigation is in left or middle softkeys.

On left softkey there can also menu called Options. On S60 there can be also smaller menu called context sensitive menu on middle softkey.

Command mapping rules

The Command mapping to softkeys follow following rules:

1. Right softkey: There can be only one "negative" Command (STOP, CANCEL, BACK, EXIT in this priority order) mapped to Right softkey, and the Command mapped there is directly invoked by softkey press.
2. Left softkey: Mutiple commands can be mapped under Left softkey in which case there is "Options" label in Left softkey and selection of it will open a menu of commands. If there's however only a single "positive" Command (OK, ITEM, SCREEN or HELP) under left softkey it will be presented directly on Left softkey. (Note: Some LCDUI components have their own operations that will be also visible under left softkey thus forcing Options menu.) If there's more than one negative Command this will force Options menu on Left softkey and the commands will be presented in the order define below.
3. Middle softkey: In Series 40 only a single context sensitive Command (OK, ITEM) is mapped to Middle softkey. In S60 multiple context sensitive Commands (OK, ITEM) can be mapped to Middle Softkey. If there's only single Command it will be shown directly in softkey, otherwise commands are visible in context sensitive menu opened from middle softkey. Normally the same commands mapped to Middle softkey are *also* available in Left softkey (directly or via Options menu). Note: Some UI components override this rule and place component specific operation directly to Middle softkey. For example, POPUP ChoiceGroup has "Open" operation in Middle softkey.

Order of Commands in menus

When Commands are presented in Options menu (or context options menu) the commands mapped to the menu are shown in following order based on the type:

1. STOP
2. OK
3. CANCEL

4. ITEM
5. SCREEN
6. HELP
7. BACK
8. EXIT.

Command labels

Command labels are used as follows:

1. If Command is direcly in softkey short Comand label is used.
2. If Command is presented in a menu (Options menu or context sensitive menu) long label is used if it fits to the space available, if not then short label is used.

Command priority property

Commands also have priority property. This is use in Nokia Command mapping only if there's multiple Commands with the same type. In this case the priority is used to choose which of the commands with same type is selected earlier in command mapping or placed before the other commands in menus.

Commands in Canvas

Commands in Canvas are mapped with the above rules. It should be noted that S60 in Canvas there is a change between 3rd ed fp 2 onward about Selection key and Commands. Prior to S60 3rd ed fp 2 Selection key in Canvas was treated as separate key that always delivered low-level key events and no commands were mapped to it. Pressing the selection key delivered -5 key code.

From 3rd ed fp2 the S60 platform is more aligned with Series 40. In Canvas the Selection key will be always mapped a Command and it will not deliver the low-level key event. This change may impact Java applications that use normal mode Canvas as they no longer get -5 key code. It also impacts applications that use Commands in full-screen mode and also rely on Selection sending low-level key event (-5). However, applications that work properly with Series 40 will work OK. There is also a compatibility mode available to get existing applications work as they were (see next section).

Most applications *that use key events from softkeys* are *full screen*Canvas applications and this change has no impact there.

S60 3rd ed fp2 will also have a text label on portrait mode like Series 40. On full screen mode all the softkeys keys including the selection key will either deliver key events or will be mapped to commands. This depends on availability of CommandListener.

Direct key events from softkeys

There's also possibility to catch low-level key events from softkeys but this is **only** possible in **full screen mode Canvas** (or Nokia UI API FullCanvas class). In Canvas, if there is no Commands and no CommandListener then softkeys will deliver normal key events. The key codes in Nokia devices are following:

Left/Top softkey: -6

Right/Bottom softkey: -7

Selection key/Middle softkey: -5

Note: make sure you don't have either Commands nor CommandListener if you want direct key events, as it depends on platform and platform version which, the non-existence of Commands or CommandListener used for checking whether to deliver key events or map to Commands.

Differences between S60 3rd ed. fp 2 and previous S60 devices

S60 3rd ed. fp 2 devices are introducing Labeled selection key to the portrait mode. This allows Java implementation to better align the Command mapping to Series 40 Java devices. Being more aligned with Series 40, however, introduces a **potential backwards incompatibility** to previous S60 devices. Since the existing user device space of previous Series 40 is larger than of S60 it was decided it is better to do this alignment as it was possible due to the introduction of labeled selection key in S60 platform.

The difference mostly is about that in Canvas the Selection key will no longer send low-level key event

(keyPressed/keyReleased/keyRepeated methods of Canvas and similar methods in CustomItem). Instead, Command gets mapped with above rules to the selection key and selection key behaves like Series 40 middle softkey. The additional benefit is that this allows Selection key to be labeled with Command's label as in native S60 3rd ed. fp 2 applications.

In full-screen Canvas if application does not set a CommandListener and no Commands to the Canvas then low-level key events will be delivered in 3rd ed fp 2 like in previous S60 devices.

There will also be a compatibility mode for S60 3rd ed fp 2 devices that allows developers to enable the legacy S60 Java Canvas Selection key behavior from 3rd ed fp 1 and earlier devices also on fp 2 devices. The compatibility mode is turned on by defining a JAD/manifest attribute:

```
Nokia-MIDlet-S60-Selection-Key-Compatibility: true
```

.

The best pattern in the applications that use normal mode Canvas is to use Commands with type ITEM. Then the applications also get them positioned to the selection key and have a proper label present in devices with labeled selection keys. For example, then these applications work well in both Series 40 and S60. For backwards compatibility to those devices without labeled selection key applications should still listen to the key event of selection key (-5 key code) and associate that key event with the same action as is activated from the selection command.

In full screen Canvas applications should not use Commands at all but always have Command listener as null and rely on low-level key events from softkeys and selection key. That way the applications work in all devices. Another reason for this is that the usability of Commands in full screen canvas is not that good.

Sample MIDlet for use of commands

```
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;

public class TestMIDlet extends MIDlet implements CommandListener {
    Display disp;

    Form form = new Form("Test Command form");

    static final Command exitCommand =  new Command("Exit", Command.EXIT, 0);

    public TestMIDlet() {
    }

    public void startApp() throws MIDletStateChangeException {
      disp = Display.getDisplay(this);
   form.addCommand(exitCommand);
   form.setCommandListener(this);
   disp.setCurrent(form);
    }

    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {
      notifyDestroyed();
    }

   public void commandAction(Command cmd,Displayable disp) {
      String cmdLabel = cmd.getLabel();
      if (cmdLabel.equals("Exit")) {
         destroyApp(true);
```

```
        }

    }
}
```