Mobile Design Pattern:ShortCuts

This design pattern is part of the Mobile Design Patterns series.

Introduction

On the mobile device, where the interaction mechanisms between the user and the application are quite hampered as compared to the desktop, it is imperative to extend the options that are available. Most devices have soft-key based interaction, where the soft-keys follow the conventions laid down by the style-guide and the same should not be tampered with. However it is surely possible to make use of the other keys which are not bound by the style-guide considerations per se to increase the ease of use for the user.

Making use of the other keys/key combinations etc to expedite the response mechanism from a user's perspective makes a lot of sense. The usability expert/UI designer should carefully consider the key functionalities of the application, and figure how many key presses it typically takes to get to those functions from the main view of the application. If they find that it is becoming tough from a user's standpoint to get to a view/function with the use of the conventional navigation approach they should think about providing some kind of **shortcuts** to the user, so that they can get there faster.

Shortcuts could be provided either by having specific hardware keys mapped to certain applications, or they can be provided through the software by programming them in such a way that they do specific tasks as dictated by the code.

Existing usage of shortcuts

Even today lots of shortcuts have been developed for the ease of the user, on any Nokia device for instance. Some of the commonly used shortcuts are as follows:-

- Application launcher shortcuts on the active idle
- Speed dial mechanism
- In case of 4 way Centre Soft Key, each side of the joystick is mapped to a specific native application.
- The LSK/RSK on the active idle provides access to customizable applications.
- Press of the Green key from the active idle invokes the call log application.
- Native gaming applications make use of keys for specific actions.
- Accessing the currently running application list by long press of task list.
- Browser shortcuts S60 web Browser shortcut keys

Extensions possible

From a usability standpoint, one can also think of providing shortcuts to the user, to make it easier for them to be able to get to a particular control/view/functionality. Some extensions possible could be as under:-

Customizing active idle shortcuts

Using the ScShortCuts Engine Wrapper API, 3rd party developers can make changes to the soft key on the active idle to display shortcuts to their applications instead of the native applications.

Details can be had from:-

ScShortcuts Engine Wrapper API

KIS000920 - Shortcuts Engine API not available in S60 3rd Edition, Feature Pack 2

Adding shortcuts to Go_To Menu

Using Symbian C++, developers can possibly look at providing shortcuts to their applications through the go_to menu.

Details can be had from:-

TSS000024 - Adding an application shortcut to the Go To menu of a S60 device

Defining shortcut keys for a Symbian app

Filter existing shortcut behavior

It is also possible to override the existing shortcut behavior. Details can be had from TSS000416 - Filtering standard S60 shortcut

(C) Copyright Nokia 2014. All rights reserved.

Custom shortcut implementation

You can provide your own custom shortcuts from within your application in the following ways :-

- On the press of a specific number combination, long/short press of a key etc.
- Password protected shortcut mechanism, in cases you don't want to display certain features/parts of the application unless the user keys in a specific number sequence. On successful keying of the key sequence the feature/view etc is accessible.
- Search field implementation on a list view, allowing the user to quickly access a specifc contact/item from the list rather than having to manually scroll to the item in question.
- Using audio cues, details of which can be found at the link **Audio Usability**
- Making use of pre-existing native functionalities by the help of the application interworking framework in the case of Symbian C++, as detailed on the link **Application Interworking(AIW)** or even using the media keys as mentioned in the link **TSS000432 - Utilising media keys**

Document them well

It is imperative that you provide details about the various shortcuts that you have implemented in your application in the context sensitive menu and the user guide. If the user is not aware of the shortcuts they might not be able to make full use of them or in some cases might end up in invoking a shortcut only to realise they did not want to use it in the first place.

Make them customizable

One should also consider providing the user a setting page where they can posssibly configure their shortcut keys with which they are more comfortable, rather then forcing them to use default combinations. Specially in the case of password enabled shortcut keys it is a must, as otherwise the user would have to memorize the default password to invoke the protected functionality.

--- Added by Mayank on 26/06/2009 ---

(C) Copyright Nokia 2014. All rights reserved.