UI test criteria

Here are some considerations to simplify some of the UI related Symbian Signed for Nokia test cases. Here are considerations of some of the most important topics of the S60 UI Style Guide; this page attempts to be a "good enough" test set to get your application through the UI test cases.

Options Menu

- Test that the first item in the options menu is the functionality that is triggered by the selection key / joystick press. This is usually "Open", "Select", "Change", etc.
- Any menu items after the first one, should be arranged by the frequency of use, most infrequent functions to the end of the list.
- Check that your options menu reacts to your applications states, meaning that there should not be any items in the options menu that cannot be triggered from the current application state. Such items need to be hidden - or they need to return a sensible error message.
- If your options menu includes several menu items, check that only the necessary options menu items are provided, when navigating forward. For example, do not provide application's About in every state, when moving forward from the main state of the application.
- "Exit" should be the last item in the options menu.

Then, a context specific options menu (a menu that concerns only one focused list item) must not contain any general options menu items (such as "Open", "Exit", etc). It should only contain the few functions that can be triggered on the selected item.

Soft keys and the selection key

- Left softkey is "OK", "Accept", or anything positive.
- Middle softkey / joystick is "Select", "Open" - this key action can never be "Back" or "Close"!
- Right softkey is "Back", "Exit", "Cancel", or anything negative.

Specified information concerning soft keys' and selection key's functionality:

- When both the left and the right softkey are labelled, the selection key must be linked to the left softkey. For example, softkey labels in queries.
- When only the left softkey is labelled, the selection key must be linked to the label and the empty right softkey must be disabled. For example, closing About information, which has OK label on the left soft key.
- When only the right softkey is labelled, the empty left soft key must be disabled and the selection key must not do the provided right soft key label’s function. For example, returning back from the application's local Help by pressing the right soft key label Back.
- NOTE! Try to prevent situations where the left soft key and the selection key trigger a different function. If possible, use options menu. For example, changing settings of the application: the left soft key label is Save and the selection key opens the focused setting to be changed -> Options menu with Change and Save items should be provided.
- NOTE! Provide the user the possibility to accept and decline, when any decision has to be done. For example, OK/Accept and Cancel/Back/Exit/Decline should be provided always in Disclaimers and in Privacy statement dialog.

Also make sure that if your application has a hierarchy of views that the right softkey acts as "Back" on other levels of the hierarchy, but as "Exit" on the main level of your application.

An exception to the above strict rule is give in the Style Guide in chapter 6.2.6.2: editable forms with options menu. In case there is a content specific options menu to an editable component, the right key can be "Done" while left is "Options".

Settings

- Make sure that the terminology used in the settings menu is consistent with the system applications. A component that consistently in other applications allows to "change" its state should not allow to "edit" in your application. This will break the total user experience.
- Make sure your settings titles are clear to understand and the settings controls intuitive to use.
- Hide settings, which do not have affect due to changing another setting. For example, GPS On / Off and GPS auto-update

interval: When the GPS is Off, the GPS auto-update interval must be hided. If the GPS is set On, the earlier defined interval should be set.

- Saving settings: settings should be saved and "activated" immediately. Ok, an example: you have an app that allows for network connection. Same app allows you to modify the IAP in use. Now after modifications the active IAP should be changed to the one selected by the user immediately (and not after an application re-start).

Navigation

- Insane vs intuitive (is it consistent with other applications).
- Simple rule: Back returns back to the same state from where the function was performed. Cancel cancels the function and returns the user safely back to the previous state without changes.
- Use indicators correctly. For example, indicate scrolling left and right only when scrolling is possible -> when music is muted the scroll left indicator must be removed and vice versa.
- By default all lists loop, and they have a scroll bar on the right hand side if the list spans over the edge of screen.
- Editable component should be in the correct format when beginning the edit operation (numeric, alphabetic).
- Editable component should not accept invalid input.
- Using UI components "by the book": not using radio buttons in a multi-selection list.

For an example:

- Using a grid component on a tabbed pane is not recommended. The grid navigation (when going over an edge) is not in-line what is expected of a tabbed pane (when clicking left or right when at the edge of the screen). Preferably, use lists on a tabbed pane.

Information notes / Query dialogs

More information in the UI Style Guide, chapter 6.7.

- information note / query dialog must reflect the state of the application accurately. As an example, imagine a situation when you have selected several entries from a multi-selection list. Now you click on "delete" and the application asks for confirmation: "Do you want to delete the selected entry?". What's wrong? You had several entries selected, how can you know which entry is being deleted by the app - or if they all will get deleted.
- As a rule of thumb any piece of information that is presented to the user should be carefully considered to have its informational content maximized ;)