Create Dynamic Settings Pages using Symbian C++

How to Create Dynamic Settings Pages

Many times there are situations where you want to create settings pages dynamically and not design them beforehand in the resource files. This is quite easy but there are some things you must take into account.

Basic Class

Your settings page class derives from **CAknSettingItemList**, defined in **aknsettingitemlist.h**. You don't have anything special to define or override but naturally you want to define your own *ConstructL()* where you can create the setting items.

Headers Required:

```
#include <aknsettingitemlist.h>
```

Library needed:

```
LIBRARY   avkon.lib
```

Settings.h

```
class CSettings : public CAknSettingItemList
{
public:
 static CSettings *NewL(CCoeControl *parent, TInt resource);
 static CSettings *NewLC(CCoeControl *parent, TInt resource);

private:
 CSettings();
 void ConstructL();
};
```

Settings.cpp

```
#include "Settings.h"

// This is used as the empty text for text setting items
_LIT(KEmptyItemText, "(none)");

CSettings *CSettings::NewL(CCoeControl *parent, TInt resource)
{
 CSettings* self = CSettings::NewLC(parent, resource);
 CleanupStack::Pop(self);
 return self;
}

CSettings *CSettings::NewLC(CCoeControl *parent, TInt resource)
{
 CSettings* self = new (ELeave)CSettings();
 CleanupStack::PushL(self);
 self->SetMopParent(parent);
 self->SetContainerWindowL(*parent);
 self->ConstructFromResourceL(resource);
 self->ConstructL();
 return self;
}

CSettings::CSettings()
{
// Here you can set the default values for your setting items
}

void CSettings::ConstructL()
{
// Here you can create the setting items
}
```

Minimal ConstructL() Without Any Items

```
void CSettings::ConstructL()
{
 TInt order = 0;
 // Get the icon array if you want to set icons
 CArrayPtr<CGulIcon>* icons =
ListBox()->ItemDrawer()->FormattedCellData()->IconArray();
 // Used for setting text items in enumerated controls
 CArrayPtr<CAknEnumeratedText>* texts;
 // Text input items
 CAknTextSettingItem *textitem;
 // Enumerated text items
 CAknEnumeratedTextPopupSettingItem *enumitem;
 // The setting item array
 CAknSettingItemArray* items;

 // Get the setting item array from the derived control
```


(C) Copyright Nokia 2013. All rights reserved.

```
 items = SettingItemArray();

 // Add your controls here to the item array

 items->RecalculateVisibleIndicesL();
 HandleChangeInItemArrayOrVisibilityL();
}
```

Adding Items to the List

There are several setting item types you can use in your settings page. They are all added to the list using **CAknSettingItemArray** which we can get using the *SettingItemArray()* method. I'll show you how to use two of the controls that I've found most useful, a text input element and an enumerated selection item.

All controls have an order number. The easiest way to keep this running is just to define an int variable that you'll increment after adding each control to the array.

After all items have been added to the array, you call *RecalculateVisibleIndicesL()* for the **CAknSettingItemArray** and *HandleChangeInItemArrayOrVisibilityL()* for your own derived class. This will ensure the control updates its state and view.

If you want to use icons for the settings, you can retrieve the icon array from the listbox using *ListBox()->ItemDrawer()->FormattedCellData()->IconArray()*.

Adding Text Input Item

Text input items are added creating an instance of **CAknTextSettingItem**. **NOTE! You MUST set the empty item text for the control even if you're not going to allow setting empty strings!** If you don't do this, the control will panic with one of the ETEXT panics when the text is empty, even while editing!

The following code can be added to the *ConstructL()* method shown earlier. **textvalue** is a descriptor defined in your class which will hold the current and modified value for the setting item.

By default, the editor won't display the OK selection if the editor is empty. If you want to allow an empty string to be input, you must uncomment the flag setting line.

```
 // Make sure the order numbers are updated
 order++;
 textitem = new (ELeave)CAknTextSettingItem(order, textvalue);
 textitem->SetEmptyItemTextL(KEmptyItemText);

 // If you want to allow setting an empty string, uncomment this
 // text->SetSettingPageFlags(CAknTextSettingPage::EZeroLengthAllowed);

 textitem->ConstructL(IsNumberedStyle(), order, _L("Setting Name"),
icons, R_TEXT_SETTING_PAGE, -1);
 items->AppendL(textitem);
```

The resource *R_TEXT_SETTING_PAGE* contains the following base settings for the item. The maximum length of the setting item is 64 and the editor will allow only text input mode.

```
RESOURCE AVKON_SETTING_PAGE r_text_setting_page
{
 type = EEikCtEdwin;
 editor_resource_id = r_text_edwin;
}

RESOURCE EDWIN r_text_edwin
{
 width = 20;
 lines = 1;
 maxlength = 64;
 default_input_mode=EAknEditorTextInputMode;
 allowed_input_modes=EAknEditorTextInputMode;
}
```

Adding Enumerated Text Item

If you want to make the user choose from a predefined set of values you can use **CAknEnumeratedTextPopupSettingItem**. It will display a similar control to the text editor but when activated will display a popup list from which the user can choose a value.

itemvalue is the *TInt* variable defined in your setting class that will hold the value for the setting item. **NOTE! You MUST set the setting value to zero when creating the control!** Otherwise the *ConstructL()* will crash. After you've added the selections to the list you can set the default value and call *LoadL()* method for the enumerated item.

You can set the selection values either in a resource file or dynamically, as we're doing here. The values are of type **CAknEnumeratedText** and have an integer value that you get as the selection value and a text string that will be shown to the user. *Note that you shouldn't use _L("asd").AllocL() ever, I'm doing this just to keep the texts in the same part of the code.*

Note that the integer values don't need to be sequential. Here we have three selections that will have integer values of 0, 1 and 10 respectively. If you set the **itemvalue** as 10, the third item will be selected. Make sure you set the **itemvalue** to a value that is present in the array.

```
 // Make sure the order numbers are updated
 order++;

 // Hold the value for the item in temp variable and set it to zero
 TInt tmpvalue = itemvalue;
 itemvalue = 0;
 enumitem = new (ELeave)CAknEnumeratedTextPopupSettingItem(order, itemvalue);
 enumitem->ConstructL(IsNumberedStyle(), order, _L("Enum item"), icons,
R_ENUMERATED_SETTING_PAGE, -1, 0, R_POPUP_SETTING_TEXTS);

 // Get the value array
 texts = enumitem->EnumeratedTextArray();
```


(C) Copyright Nokia 2013. All rights reserved.

```
 // Empty it
 texts->ResetAndDestroy();
 texts->AppendL(new CAknEnumeratedText(0, _L("first (0)").AllocL()));
 texts->AppendL(new CAknEnumeratedText(1, _L("second (1)").AllocL()));
 texts->AppendL(new CAknEnumeratedText(10, _L("third (10)").AllocL()));

 // Set the real value for the item
 itemvalue = tmpvalue;
 // Tell the control to load the value in
 enumitem->LoadL();
 items->AppendL(enumitem);
```

The resources **R_ENUMERATED_SETTING_PAGE** and **R_POPUP_SETTING_TEXTS** used in the code contain the following settings. We have a dummy item in the text array which we'll clear away when we're creating the control.

```
RESOURCE AVKON_SETTING_PAGE r_enumerated_setting_page
{
 type = EAknCtPopupSettingList;
 editor_resource_id = r_popup_setting_list;
}

RESOURCE POPUP_SETTING_LIST r_popup_setting_list
{
 flags = 0;
}

RESOURCE AVKON_POPUP_SETTING_TEXTS r_popup_setting_texts
{
 setting_texts_resource = r_texts;
}

RESOURCE ARRAY r_texts
{
 items =
 {
  AVKON_ENUMERATED_TEXT { value=0; text = "dummy";}
 };
}
```

Constructing the Setting Page

Resources Needed

When you're using the setting page from your code, you must have a resource for it that contains the basic settings. Here we only have a title for tha page.

```
RESOURCE AVKON_SETTING_ITEM_LIST r_settings_list
{
 title = "Settings";
}
```

Construction

```
 CSettings settings = CSettings::NewLC(this, R_SETTINGS_LIST);
```

Loading And Saving Settings Values

When you want to save the setting item values, you must first call *SaveSettingsL()*. This will call the same method on all the setting items and instruct them to store their values in the variables you defined when you created them. After this you can save the values.

If you want to load the values after the controls have been created, you can use *LoadSettingsL()* which in turn will call the same method on all the setting items and instructs them to load the values from the variables you defined.

Note! You can use one *TInt* variable and one descriptor for all your setting items if you want to (sometimes the dynamically created setting items are totally dynamic and you don't know how many there are etc). If you want to do this, you must call *SaveSettingsL()* for all your setting items one by one and save the values before calling it for the next.

Possible Panics

Setting item lists can generate several panics. The most annoying are Setting Item List panics (which will show as Setting Item Lis X on your device). They are not documented in official documentation but I have written about them here to help others that have problems with them. Also ETEXT panics are not uncommon with text items.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.