



## Contents

- 1 Using CEikEdwin Text Editor
  - ◆ 1.1 What Is CEikEdwin?
  - ◆ 1.2 Instantiating
  - ◆ 1.3 Setting Text on the Editor
  - ◆ 1.4 Retrieving Text from the Editor
  - ◆ 1.5 Deleting Parts of the Text

## Using CEikEdwin Text Editor

### What Is CEikEdwin?

If you want to display text on the screen and let the user scroll around it or even edit it, you want to use **CEikEdwin**. This is a simple text editor control that doesn't have any fancy coloring or font changing. If you need those features, use **CRichText** based controls.

### Instantiating

To create a **CEikEdwin** control you need to instantiate it from a resource. Here we create an editor that has a maximum length of 511, allows any input mode and defaults to text input. **Note!** If you don't set the **special\_character\_table** correctly, the devices won't show anything when pressing the special character key. You can also set the table to show a nemail or web related character set, when needed.

You may also want to set some flags. These can be set with the *flags* resource item. For example, if you want to create a read-only text editor control, you may want to set the flags to *EEikEdwinNoHorizScrolling* | *EEikEdwinReadOnly* | *EEikEdwinNoAutoSelection*

```
RESOURCE EDWIN r_inputbox
{
    maxlength = 511;
    default_case = EAknEditorTextCase;
    allowed_case_modes = EAknEditorAllCaseModes;
    numeric_keymap = EAknEditorStandardNumberModeKeymap;
    default_input_mode = EAknEditorTextInputMode;
    allowed_input_modes = EAknEditorAllInputModes;
    special_character_table = R_AVKON_SPECIAL_CHARACTER_TABLE_DIALOG;
}
```

```
CEikEdwin* iInputbox;

// Instantiate the control
iInputbox = new (ELeave)CEikEdwin;
```

## Using\_CEikEdwin\_Text\_Editor

```
iInputbox->SetContainerWindowL(*this);

// Create a resource reader that we'll use to get the settings
TResourceReader reader;
iEikonEnv->CreateResourceReaderLC(reader, R_INPUTBOX);
iInputbox->ConstructFromResourceL(reader);
CleanupStack::PopAndDestroy(); // reader
```

## Setting Text on the Editor

Usually you'll want to set the text displayed on the control. This can be done by using the *SetTextL()* method. If you want to set the cursor position, you can call the *SetCursorPosL()* method.

```
iInputbox->SetTextL(text);

// Set the cursor to the beginning of the text and not do any selecting
iInputbox->SetCursorPosL(0, EFalse);
```

## Retrieving Text from the Editor

If you want to retrieve the text that's written in the editor, you can get it as *HBufC\** or copy the text to a descriptor. Usually it's easier to get it as *HBufC* since you don't have to worry about how much space you need.

```
HBufC *text = NULL;

text = iInputbox->GetTextInHBufL();
//Must check text, GetTextInHBufL() will return NULL if no text is set/entered in editor.
if ((text != NULL) && (text->Length() > 0))
{
// There is some text, do what you want with it
}

// Don't forget to delete the object afterwards!
delete text;
```

## Deleting Parts of the Text

If you want to delete parts of the text that's written in the editor, you can use the instance of **CPlainText** object that can be retrieved from the control.

If you don't do the deletion right, you'll get some of the [ETEXT Panics](#) that aren't documented. So be careful to do it exactly as shown here. The most important thing is the **HandleTextChangedL()** call.

```
CPlainText *text;

// Get the instance of CPlainText from the control
text = iInputbox->Text();

// Delete the first 10 characters in the editor
TInt start = 0;
TInt length = 10;
```

## Using\_CEikEdwin\_Text\_Editor

```
// NOTE! If there are less than 10 characters in the editor,  
//      the deletion will panic. This is why we check the length  
  
if (text->DocumentLength() < start + length)  
    =length>DocumentLength() - start;  
  
// Cannot delete since length is below zero or start is beyond the end of text  
if ((length <= 0) || (start >= text->DocumentLength()))  
    return;  
  
// Delete the characters  
text->DeleteL(start, length);  
// After that, you MUST immediately inform the CEikEdwin that the text  
// has changed, otherwise you'll get an ETEXT panic!  
iInputbox->HandleTextChangedL();  
// Clear possible selection in th
```