

## Editor\_example

**C**EditorContainer illustrates a simple container which you can use to add an editor into your project.

To modify the font used with the editor you can select another font in the *CreateEditorL()* method instead of *LatinBold16()*. The font color can be set by changing the **KRgbBlack** constant used in the **TCharFormat**.

Note that using a rich text editor can cause wasted resources if you don't need different paragraph or character formatting. In these cases you should use **CEikEdwin** which will allow you to set the font and color for the whole text. Also you could use **CEikRichTextEditor** directly or derive from it rather than using a container within a container.

## Contents

- [1 Link against:](#)
- [2 Header Required:](#)
- [3 Required capabilities:](#)
- [4 Editor Container.cpp](#)
- [5 Editor Container.h](#)

## Link against:

```
avkon.lib eikcoctl.lib eikctl.lib form.lib uiklaf.lib
```

## Header Required:

```
#include <eikrted.h>
```

## Required capabilities:

None

## Editor\_Container.cpp

```
#include<Editor_container.h>
void CEditorContainer::ConstructL(void)
{
    Creat@WindowL
```

## Editor\_example

```
(SEikEnv::Static()->EikAppUi()->ClientRect());

CreateEditorL

ActivateL
}

void CEditorContainer::CreateEditorL(void)
{
    #ifdef ELeave
    CEikRichTextEditor;
    ->EikEditorFlags(EAknEditorFlagEnableScrollBars);
    ->EikContainerWindowL(*this);
    ->EikAknWrap(ETrue);
    ->EikConstructL(this, 0, 0, 0);

    ->EikFocus(ETrue);
    ->EikAknCcpuSupportL(ETrue);
    ->EikCursorPosL(0, EFalse);
    ->EikAknWrap(ETrue);

    TFontSpec fontSpec(EFontSpecInTwips());
    TCharFormat charFormat(fontSpec.iTypeface.iName, fontSpec.iHeight);
    TCharFormatMask charFormatMask

    charFormat.iTextColor = KRgbBlack;
    charFormat.iColor(EAttColor);

    charFormat.iTypeface(EAttFontTypeface);
    charFormat.iHeight(EAttFontHeight);
    ->ApplyCharFormatL(charFormat, charFormatMask);

    ->EikAknScrollBarsL();
    SizeChanged
    }

CEditorContainer::~CEditorContainer()
{
    delete iEditor;
}

void CEditorContainer::HandleResourceChange(TInt aType)
{
    TRect rect

    if ( aType==KEikDynamicLayoutVariantSwitch )
    {
        :ApplyMethodL(EikAknLayoutUtils::EMainPane, rect );
        (rect); SetRect
    }

    CCoEHandleResourceChange(aType);
}

void CEditorContainer::SizeChanged()
{
    TRect rect;

    if(iEditor)
    {
#ifdef __SERIES60_3X__

```

## Editor\_example

```

    TRect $EditorBarRect;
    TRect $ScrollBarRect;
    TRect $EditorBarFrame() ->
VerticalScrollBar() -> Rect();
    -> SetExtentPoint(0, StatusLineHeight),
    (rect.Width() + $ScrollBarRect.Width(), rect.Height() - StatusLineHeight));
#else
    -> SetExtentPoint(0, StatusLineHeight),
    (rect.Width() + $ScrollBarRect.Width(), rect.Height() - StatusLineHeight));
#endif
}
}

TInt CEditorContainer::CountComponentControls() const
{
    if (iEditor)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}

CCoeControl* CEditorContainer::ComponentControl(TInt /*aIndex*/) const
{
    return iEditor;
}

void CEditorContainer::Draw(const TRect& aRect) const
{
    CWindowGSystemGc();
    ClearGc();
}

TKeyResponse CEditorContainer::OfferKeyEventL(const TKeyEvent& aKeyEvent,
TEventCode aType)
{
    TKeyResponse Ret(TKeyResNotConsumed);

    if (iEditor)
    {
        Ret = iEditor->OfferKeyEventL(aKeyEvent, aType);
    }

    return Ret;
}

void CEditorContainer::SetEditorTextL(const TDesC& aText)
{
    if (iEditor)
    {
        iEditor->ClearSelectionL();
        iEditor->RichTextEditorReset();

        iEditor->RichTextEditorInsertL(0, aText);
        iEditor->RichTextEditorInsertL(iEditor->RichTextEditorLength(),
CEditableText::ELineBreak);

        iEditor->UpdateScrollBarL();
        iEditor->DrawNow();
    }
}

```

**Editor\_Container.h**

```

class CEditorContainer : public CCoeControl
{
public:
    void ConstructL(void);
    ~CEditorContainer();
    void SetEditorTextL(const TDesC& aText);
public:
    TInt CountComponentControls() const;
    CCoeControl* ComponentControl(TInt aIndex) const;
    TKeyResponse OfferKeyEventL(const TKeyEvent& aKeyEvent, TEventCode aType);
private:
    void CreateEditorL(void);
    virtual void SizeChanged();
virtual void HandleResourceChange(TInt aType);
    void Draw(const TRect& aRect) const;
private:
    CEikEdhTextEditor* Editor;
};

```