Using scrollbars in Symbian container control

Overview

This code snippet shows how to add scrollbars to a container control that has custom controls and how to show only visible components and search for more of them in view top and bottom.

This example extends the existing code snippet Changing the focus of a Symbian custom control. Check the See also section of this article for other snippets of the Custom Control series.

CMyContainerControl - Header

Add these methods and member data to the `CMyContainerControl` component header.

`MoveFocusUpL()` Moves focus to ask for more components to the screen at the bottom of the view
`MoveFocusDownL()` Moves focus to ask for more components to the screen at the bottom of the view
`UpdateScrollBarFrameL()` Updates scroll bar
`ShowNextItem()` Searches for more items to the screen at the bottom of the view
`ShowPrevItem()` Searches for more items to the screen at the top of the view

```
#include <eiksbfrm.h>

private:
    void ShowNextItem();
    void ShowPrevItem();
    void MoveFocusUpL();
    void MoveFocusDownL();
    void UpdateScrollBarFrameL();
private:
    CEikScrollBarFrame*                  iScrollBarFrame;
    TAknDoubleSpanScrollBarModel iHDsSbarModel;
    TAknDoubleSpanScrollBarModel iVDsSbarModel;
    TInt iFocusedIndex;
```

CMyContainerControl - Source

Create scroll bar in `CMyContainerControl::ConstructL()`

```
// Creating Scroll Bars
iScrollBarFrame = new ( ELeave ) CEikScrollBarFrame( this, NULL );
iScrollBarFrame->CreateDoubleSpanScrollBarsL( ETrue, EFalse );
iScrollBarFrame->SetTypeOfVScrollBar( CEikScrollBarFrame::EDoubleSpan );
iScrollBarFrame->SetScrollBarVisibilityL( CEikScrollBarFrame::EOff,
CEikScrollBarFrame::EOff );
```

Update scroll bar data in `CMyContainerControl::UpdateScrollBarFrameL()` The method will be called in `CMyContainerControl::AddControlL()` after adding components into this container control.

```
void CMyContainerControl::UpdateScrollBarFrameL()
    {
    TInt controlsHeight = 0; // Height of the controls
    TInt height = 0; // This view height

    // Calculate components height
    CCoeControlArray::TCursor cursor = Components().Begin();
    CCoeControl* ctrl = NULL;
    while ((ctrl = cursor.Control<CCoeControl>()) != NULL)
        {
        controlsHeight += ctrl->MinimumSize().iHeight;
        cursor.Next();
        }

    // This view height
    height = Rect().Height();

    // Set teh scrollbar visible if fields do not fit the screen
    if( controlsHeight > height &&
        iScrollBarFrame->VScrollBarVisibility() ==
        CEikScrollBarFrame::EOff)
        {
        iScrollBarFrame->SetScrollBarVisibilityL(
                CEikScrollBarFrame::EOff, // horizontal
                CEikScrollBarFrame::EOn); // vertical
        }
    // Hide the scrollbar if fields fit the screen
    else if ( controlsHeight <= height &&
              iScrollBarFrame->VScrollBarVisibility() ==
              CEikScrollBarFrame::EOn)
        {
        iScrollBarFrame->SetScrollBarVisibilityL(
                CEikScrollBarFrame::EOff, // horizontal
                CEikScrollBarFrame::EOff); // vertical
```


(C) Copyright Nokia 2013. All rights reserved.

```
        }

    // Update scroll bar position
    iVDsSbarModel.SetScrollSpan(Components().Count());
    iVDsSbarModel.SetWindowSize(1);
    iVDsSbarModel.SetFocusPosition(iFocusedIndex);
    TEikScrollBarFrameLayout layout;
    layout.iTilingMode = TEikScrollBarFrameLayout::EInclusiveRectConstant;
    TRect rect = Rect();
    iScrollBarFrame->TileL(&iHDsSbarModel,&iVDsSbarModel,rect,rect,layout);
    iScrollBarFrame->SetVFocusPosToThumbPos(iVDsSbarModel.FocusPosition());
    }
```

`CMyContainerControl::MoveFocusUpL()` moves active component focus and asks to show more components if the cursor reaches the screen top.

```
void CMyContainerControl::MoveFocusUpL()
    {
    CCoeControlArray::TCursor cursor = Components().Begin();
    CCoeControl* ctrl = NULL;
    CCoeControl* prevCtrl = NULL;
    while ((ctrl = cursor.Control<CCoeControl>()) != NULL)
        {
        if (ctrl->IsFocused())
            {
            if (prevCtrl)
                {
                // Set focus to previous control
                ctrl->SetFocus(EFalse);
                prevCtrl->SetFocus(ETrue);
                iFocusedIndex--;

                // Focus is over the view?
                if (iFocusedIndex > 0 && !prevCtrl->IsVisible())
                    {
                    ShowPrevItem();
                    }
                break;
                }
            else
                {
                break; // First control is already focused
                }
            }
        prevCtrl = ctrl;
        cursor.Next();
        }

    UpdateScrollBarFrameL();
    }
```

`CMyContainerControl::MoveFocusDownL()` moves active component focus and asks to show more components if the cursor reaches the screen bottom.

```
void CMyContainerControl::MoveFocusDownL()
    {
    CCoeControlArray::TCursor cursor = Components().Begin();
    CCoeControl* ctrl = NULL;
    CCoeControl* nextCtrl = NULL;
    while ((ctrl = cursor.Control<CCoeControl>()) != NULL)
        {
        if (ctrl && ctrl->IsFocused())
            {
            cursor.Next();
            nextCtrl = cursor.Control<CCoeControl>();
            if (nextCtrl)
                {
                // Set focus to next control
                ctrl->SetFocus(EFalse);
                nextCtrl->SetFocus(ETrue);
                iFocusedIndex++;

                // Focus is over the view?
                if (!nextCtrl->IsVisible())
                    {
                    ShowNextItem();
                    }
                break;
                }
            else
                {
                break; // Last control is already focused
                }
            }
        cursor.Next();
        }

    UpdateScrollBarFrameL();
    }
```

`void CMyContainerControl::ShowNextItem()` and `void CMyContainerControl::ShowPrevItem()` find more components to view and set them to the correct position.

```
void CMyContainerControl::ShowNextItem()
    {
    TPoint position;

    // Goes throught all components of this container control
    CCoeControlArray::TCursor cursor = Components().Begin();
    CCoeControl* ctrl = NULL;
    TBool focusedFound = EFalse;
    while ((ctrl = cursor.Control<CCoeControl>()) != NULL)
        {
        if (ctrl && ctrl->IsFocused())
            {
            focusedFound = ETrue;
            }

        if (focusedFound)
            {
            // Set size
```


(C) Copyright Nokia 2013. All rights reserved.

```
            TSize size = ctrl->MinimumSize();
            size.SetSize(Rect().Width(),size.iHeight);
            ctrl->SetSize(size);

            // Does it fit to the screen?
            if ((position.iY + size.iHeight) >= Rect().iBr.iY)
                {
                ctrl->MakeVisible(EFalse);
                focusedFound = EFalse; // Let rest components be MakeVisible(EFalse)
                }
            else
                {
                ctrl->MakeVisible(ETrue);

                // Set position
                ctrl->SetPosition(position);

                // Store position of last component
                position.iY += size.iHeight;
                }
            }
        else
            {
            ctrl->MakeVisible(EFalse);
            }
        cursor.Next();
        }
    }


void CMyContainerControl::ShowPrevItem()
    {
    TPoint position;
    position.iY = Rect().iBr.iY;

    // Goes throught all components from the last -> to the first one
    CCoeControlArray::TCursor cursor = Components().End();
    CCoeControl* ctrl = NULL;
    TBool focusedFound = EFalse;
    while (cursor.Prev())
        {
        ctrl = cursor.Control<CCoeControl>();
        if (ctrl && ctrl->IsFocused())
            {
            focusedFound = ETrue;
            }

        if (focusedFound)
            {
            // Set size
            TSize size = ctrl->MinimumSize();
            size.SetSize(Rect().Width(),size.iHeight);
            ctrl->SetSize(size);

            // Fix position
            position.iY -= size.iHeight;

            if ((position.iY) <= Rect().iTl.iY-2)
                {
                // Does not fit anymore
                ctrl->MakeVisible(EFalse);
                focusedFound = EFalse; // Let rest components be MakeVisible(EFalse)
                }
            else
                {
                ctrl->MakeVisible(ETrue);

                // Set position
                ctrl->SetPosition(position);
                }
            }
        else
            {
            ctrl->MakeVisible(EFalse);
            }
        }
    }
```

Postconditions

Custom control has scrollbars that show the position of the focus in the view. More components will be shown when reaching the bottom or top of the view.

See also

Custom Control Series:

- How to create a custom control in Symbian C++ How to define custom control
- Constructing a Symbian custom control from a resource Creating a control from a resource
- Constructing a Symbian container control Creating a container control
- Changing the focus of a Symbian custom control Handling key events and changing active custom control focus
- Using a Symbian custom control in a dialog Adding a custom control into CAknDialog
- File:CustomControl.zip Example code patch

COPY-PASTE the desired platforms into the wiki text and save. All relevant version categories should be applied.
Windows Phone: [[Category:Windows Phone]]
[[Category:Windows Phone 7.5]]
[[Category:Windows Phone 8]]

Series 40: [[Category:Series 40]]
[[Category:Series 40 1st Edition]] [[Category:Series 40 2nd Edition]]

(C) Copyright Nokia 2013. All rights reserved.

[[Category:Series 40 3rd Edition (initial release)]] [[Category:Series 40 3rd Edition FP1]] [[Category:Series 40 3rd Edition FP2]]
[[Category:Series 40 5th Edition (initial release)]] [[Category:Series 40 5th Edition FP1]]
[[Category:Series 40 6th Edition (initial release)]] [[Category:Series 40 6th Edition FP1]] [[Category:Series 40 Developer Platform 1.0]] [[Category:Series 40 Developer Platform 1.1]] [[Category:Series 40 Developer Platform 2.0]]

Symbian: [[Category:Symbian]]
[[Category:S60 1st Edition]] [[Category:S60 2nd Edition (initial release)]] [[Category:S60 2nd Edition FP1]] [[Category:S60 2nd Edition FP2]]
[[Category:S60 2nd Edition FP3]]
[[Category:S60 3rd Edition (initial release)]] [[Category:S60 3rd Edition FP1]] [[Category:S60 3rd Edition FP2]]
[[Category:S60 5th Edition]]
[[Category:Symbian^3]] [[Category:Symbian Anna]] [[Category:Nokia Belle]]

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.