Avoid flickering with double buffering on Symbian

Overview

When images are drawn directly on the screen and there is an animation, the screen may flicker. A common technique to solve this is to draw images on an off-screen buffer, and then copy its contents to the screen when the drawing operations are finished.

The following example shows how to implement this technique. Two buffers are needed, hence the term "double buffering". This off-screen buffer has the same screen dimensions as the original one.

MMP file

The following capabilities and libraries are required:

```
CAPABILITY NONE
LIBRARY fbscli.lib
LIBRARY bitgdi.lib
```

Header

Here is an example header of the CCoeControl component where double buffering is enabled.

```
#include <coecntrl.h>
#include <bitstd.h>
#include <bitdev.h>

class CMyContainer : public CCoeControl
    {
    public:
        void NewL(const TRect& aRect)
        void ConstructL(const TRect& aRect);
        virtual ~CImageConverterContainer();
    private:
        void CMyContainer();
        void SizeChanged();
        void HandleResourceChange(TInt aType);
        TInt CountComponentControls() const;
        CCoeControl* ComponentControl(TInt aIndex) const;
        void Draw(const TRect& aRect) const;

    // for Double buffering
    public:
        void DrawToBackBuffer();
    // for Double buffering
    private:
        void CreateBackBufferL();
        void ReleaseBackBuffer();

    // for Double buffering
    private:
        CFbsBitmap*                     iBackBuffer;
        CFbsBitmapDevice*               iBackBufferDevice;
        CFbsBitGc*                      iBackBufferContext;
        TSize iBackBufferSize;
```


(C) Copyright Nokia 2014. All rights reserved.

```
    // other member variables of the class if needed
    private:
    };
```

Source

This source code is only for data related to double buffering:

```
void CMyContainer::SizeChanged()
    {
    // Delete back buffer and create a new one with new size
    ReleaseBackBuffer();
    CreateBackBufferL();
    DrawToBackBuffer();
    }

void CMyContainer::CreateBackBufferL()
    {
    // Create back buffer bitmap
    iBackBuffer = new (ELeave) CFbsBitmap;

    User::LeaveIfError( iBackBuffer->Create(Size(),
                        iEikonEnv->DefaultDisplayMode()));

    // Create back buffer graphics context
    iBackBufferDevice = CFbsBitmapDevice::NewL(iBackBuffer);
    User::LeaveIfError(iBackBufferDevice->CreateContext(iBackBufferContext));
    iBackBufferContext->SetPenStyle(CGraphicsContext::ESolidPen);

    iBackBufferSize = iBackBuffer->SizeInPixels();
    }

void CMyContainer::ReleaseBackBuffer()
    {
    // Release double buffering classes
    if (iBackBufferContext)
        {
        delete iBackBufferContext;
        iBackBufferContext = NULL;
        }
    if (iBackBufferDevice)
        {
        delete iBackBufferDevice;
        iBackBufferDevice = NULL;
        }
    if (iBackBuffer)
        {
        delete iBackBuffer;
        iBackBuffer = NULL;
        }
    iBackBufferSize = TSize(0, 0);
    }

void CMyContainer::DrawToBackBuffer()
    {
    if (!iBackBufferContext)
```


(C) Copyright Nokia 2014. All rights reserved.

```
        {
        return;
        }

    iBackBufferContext->Clear();

    // TODO: do your drawing there.
    // Remember to draw to iBackBufferContext buffered graphic contex

    // Example of drawing bitmap into iBackBufferContext
    //if(iBitmap)
    //    {
    //    iBackBufferContext->BitBlt( iPicturePoint, iBitmap );
    //    }
    }

void CMyContainer::Draw(const TRect& aRect) const
    {
    CWindowGc& gc = SystemGc();

    // Copy backbuffer to the screen
    gc.BitBlt(TPoint(0, 0), iBackBuffer);

    // TODO: Remove your all drawing from there to DrawToBackBuffer
    }

CMyContainer::~CMyContainer()
    {
    ReleaseBackBuffer();
    }
```

Using

Previously this was done as follows:

```
iAppContainer->DrawNow();
```

and now with double buffering:

```
iAppContainer->DrawToBackBuffer();
iAppContainer->DrawNow();
```

Postconditions

The screen does not flicker any more.

See also

Anti-tearing with CDirectScreenBitmap

Double buffering in Wikipedia

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.