Implementing autofocus functionality (S60 3rd Edition, pre-FP2)

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

The article is believed to be still valid for the original topic scope.

Overview

This snippet demonstrates how to implement autofocus (AF) functionality in S60 3rd Edition devices prior to S60 3rd Edition, FP2.

Note: In order to use the code, you need to install the AF extension for your SDK. You can obtain it from the S60 Platform: Camera Example with AutoFocus Support. Read the release notes of the example for more information.

MMP file

This snippet requires the following capabilities and libraries:

```
CAPABILITY UserEnvironment
```

```
LIBRARY ecam.lib

#ifdef WINSCW
    LIBRARY CamAutoFocus.lib
#else
    STATICLIBRARY CamAutoFocus_s.lib
#endif
```

Header file: Camera.hrh

```
enum TEngineState
    {
    EIdle,
    EFocusing
    };
```

Header file: CCameraEngine.h

```
#include <CCamAutoFocus.h>  // CCamAutoFocus, MCamAutoFocusObserver
#include <ECam.h>  // CCamera, MCameraObserver

class CCameraEngine : public MCameraObserver,
                      public MCamAutoFocusObserver
    {
    // Constructors and destructors omitted for brevity
    // ...

    private:  // Methods from base classes
        /**
         * From MCamAutoFocusObserver.
         * Gets called when CCamAutoFocus::InitL() is completed.
         */
         virtual void InitComplete(TInt aError);

        /**
         * From MCamAutoFocusObserver.
         * Gets called when CCamAutoFocus::AttemptOptimisedFocusL() is
         * completed.
         */
         virtual void OptimisedFocusComplete(TInt aError);

         // Methods from MCameraObserver omitted for brevity
         // ...

    private:  // New functions
        /**
         * Starts the optimised autofocus operation.
         * Does nothing if AF is not supported.
         */
        void StartFocusL();

        /**
         * Cancels an ongoing autofocus operation.
         */
        void CancelFocus();

    private:  // Data
        CCamera* iCamera;
        CCamAutoFocus* iAutoFocus;
        TEngineState iState;
```


(C) Copyright Nokia 2013. All rights reserved.

```
    };
```

Source file: CCameraEngine.cpp

```
#include "CCameraEngine.h"

CCameraEngine::CCameraEngine(/* Parameters omitted */) :
    iAutoFocus(0)
    {
    }

void CCameraEngine::ConstructL()
    {
    // TODO: It is assumed here that the device has a camera.
    // Add error handling if this may not be the case.

    // Camera index 0 is the main camera
    iCamera = CCamera::NewL(*this, 0);

    // Try to create the autofocus object
    TRAPD(afErr, iAutoFocus = CCamAutoFocus::NewL(iCamera));
    if (afErr)
        {
        // TODO: Error handling: KErrExtensionNotSupported
        iAutoFocus = 0;
        }
    }

// Other constructors omitted for brevity
// ...

CCameraEngine::~CCameraEngine()
    {
    delete iAutoFocus;
    delete iCamera;
    }

// From MCameraObserver. Gets called when CCamera::PowerOn() is completed.
void CCameraEngine::PowerOnComplete(TInt aError)
    {
    // TODO: Error handling (aError)

    // Try to init the AF control - not fatal if fails
    if (iAutoFocus)
        {
        TRAPD(afErr, iAutoFocus->InitL(*this));
        if (afErr)
            {
            // TODO: Error handling: KErrExtensionNotSupported
            delete iAutoFocus;
            iAutoFocus = 0;
            }
        }
    }

void CCameraEngine::InitComplete(TInt aError)
    {
    // TODO: Error handling (aError)
    }

void CCameraEngine::StartFocusL()
    {
    // Do nothing if autofocus is not supported or focusing is already in
    // progress
    if (!iAutoFocus || iState == EFocusing)
        return;

    // Set AF range to normal before first focus attempt
    iAutoFocus->SetFocusRangeL(CCamAutoFocus::ERangeNormal);

    // Attempt focusing. Calls OptimisedFocusComplete when ready.
    iState = EFocusing;
    iAutoFocus->AttemptOptimisedFocusL();
    }

void CCameraEngine::OptimisedFocusComplete(TInt aError)
    {
    iState = EIdle;

    if (!aError)
        {
        // Play a sound etc.
        }
    }

void CCameraEngine::CancelFocus()
    {
    // Do nothing if autofocus is not supported or focusing is not in
    // progress
    if (!iAutoFocus || iState != EFocusing)
        return;

    iState = EIdle;
    iAutoFocus->Cancel();
    }
```

Header file: CCameraController.h

```
#include <e32base.h>

// Forward declarations
class CCameraEngine;

class CCameraController : public CBase
    {
    // ...

    public:  // New functions
        /**
         * Starts the optimized autofocus operation.
```


(C) Copyright Nokia 2013. All rights reserved.

```
         */
        void StartFocusL();

        /**
         * Cancels an ongoing autofocus operation.
         */
        void CancelFocus();

    private:  // Data
        CCameraEngine* iCameraEngine;
    };
```

Source file: CCameraController.cpp

```
#include "CCameraEngine.h"

// ...

void CCameraController::StartFocusL()
    {
    iCameraEngine->StartFocusL();
    }

void CCameraController::CancelFocus()
    {
    iCameraEngine->CancelFocus();
    }
```

Header file: CCameraAppUi.h

```
#include <aknappui.h>

const TInt KStdKeyCameraFocus = 0xE2;

class CCameraExampleAppUi : public CAknAppUi
    {
    // ...

    private: // Functions from base classes
        /**
         * From CEikAppUi.
         * Handles key events.
         *
         * @param aKeyEvent Event to be handled.
         * @param aType Type of the key event.
         * @return Response code (EKeyWasConsumed, EKeyWasNotConsumed).
         */
        virtual TKeyResponse HandleKeyEventL(const TKeyEvent& aKeyEvent,
                TEventCode aType);
    };
```

Source file: CCameraAppUi.cpp

```
TKeyResponse CCameraAppUi::HandleKeyEventL(const TKeyEvent& aKeyEvent,
        TEventCode aType)
    {
    // Handle autofocus request
    if (aKeyEvent.iScanCode == KStdKeyCameraFocus)
        {
        // Obtain the engine state from CCameraController
        // TEngineState state = ...

        if (state != EIdle && state != EFocusing)
            {
            return EKeyWasConsumed;
            }

        // When the focus key is pressed down, start the focus operation.
        // When it is released, cancel the ongoing focus operation.
        switch (aType)
            {
            case EEventKeyDown:
                iCameraController->StartFocusL();
                break;
            case EEventKeyUp:
                iCameraController->CancelFocus();
                break;
            default:
                break;
            }
        return EKeyWasConsumed;
        }

    return EKeyWasNotConsumed;
    }
```

Postconditions

The camera focuses when the focus key is pressed.

See also

- Archived:Displaying viewfinder data using ECam
- S60 Platform: Camera Example with AutoFocus Support

(C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.