# Showing a global progress dialog using CAknGlobalProgressDialog

**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 code snippet shows how to create a global progress dialog `CAknGlobalProgressDialog`. A global progress dialog is constructed by calling the `NewL()` method and is executed using the `ShowProgressDialogL()` method. An active object has to be constructed to handle the dismissal of the dialog. This example uses the generic class `CGlobalQueryHandlerAO` that is also used with other global query code snippets. The class `CGlobalQueryContainer` is used to launch the query and it is also the observer that implements the interface `MGlobalQueryObserver`.

This snippet can be self-signed.

## MMP file

The following libraries are required:

```
LIBRARY cone.lib            //CEikonEnv, CCoeEnv
LIBRARY avkon.lib           //Avkon resources
LIBRARY CommonEngine.lib    //StringLoader
LIBRARY euser.lib           //CActive, CleanupStack, CPeriodic
```

## Resource file

.rss

```
#include <eikon.rh>
#include <avkon.rsg>
#include <avkon.rh>

//...
RESOURCE TBUF r_global_query_progress_text
    {
    buf = "<progress dialog text here>";
    }
```

## Header files

- observer interface

```
#ifndef GLOBALQUERYOBSERVER_H
#define GLOBALQUERYOBSERVER_H

class MGlobalQueryObserver
    {
public:
    //keep these methods short running because they are called
    //from within an active object
    virtual void ProcessOkOptionL() = 0;
    virtual void ProcessCancelOptionL() = 0;
    virtual void ProcessDoneOptionL() = 0;
    virtual void ProcessSelectedItemL(const TInt aItem) = 0;
    };

#endif  // GLOBALQUERYOBSERVER_H
```

- handler active object

```
#ifndef GLOBALQUERYHANDLERAO_H
#define GLOBALQUERYHANDLERAO_H

#include <e32base.h>  //CActive
#include "GlobalQueryObserver.h"

class CGlobalQueryHandlerAO : public CActive
    {
public:
    ~CGlobalQueryHandlerAO();
    static CGlobalQueryHandlerAO* NewL(MGlobalQueryObserver& aObserver);
    static CGlobalQueryHandlerAO* NewLC(MGlobalQueryObserver& aObserver);
public:
    void StartHandler();
```


(C) Copyright Nokia 2013. All rights reserved.

```
private:
    CGlobalQueryHandlerAO(MGlobalQueryObserver& aObserver)
            : CActive (EPriorityStandard), iObserver(aObserver){};
    void ConstructL();
private: // from CActive
    void RunL();
    void DoCancel();
private: // data
    MGlobalQueryObserver& iObserver;
    };
#endif //GLOBALQUERYHANDLERAO_H
```

container / observer

```
#ifndef GLOBALQUERYCONTAINER_H
#define GLOBALQUERYCONTAINER_H

#include <coecntrl.h> // CCoeControl
#include "GlobalQueryObserver.h"
#include <aknglobalprogressdialog.h> //CAknGlobalProgressDialog

const TInt KFinalExampleStep = 100;

class CGlobalQueryHandlerAO;

class CGlobalQueryContainer : public CCoeControl, MGlobalQueryObserver
    {
public:
//...
void MakeExampleQueryL();
private:
//...
    static TInt TimerCallback(TAny* aThis);
    void GlobalExampleProcessTimerEvent();
public: // from MGlobalQueryObserver
    void ProcessOkOptionL(); //not used
    void ProcessCancelOptionL();
    void ProcessDoneOptionL();
    void ProcessSelectedItemL(const TInt aItem); //not used
//...
private:
    CGlobalQueryHandlerAO*       iQueryHandlerAO;
    CAknGlobalProgressDialog*    iGlobalProgressDialog;
    CPeriodic*                   iTimer;
    TInt iCurrentExampleStep;
    };
#endif  // GLOBALQUERYCONTAINER_H
```

## Source files

handler active object

```
#include "GlobalQueryHandlerAO.h"
#include <avkon.hrh> //EAknSoftkeyOk, EAknSoftkeyCancel, EAknSoftkeyDone

CGlobalQueryHandlerAO::~CGlobalQueryHandlerAO() { Cancel(); }

void CGlobalQueryHandlerAO::ConstructL()
    {
    CActiveScheduler::Add(this);
    }

CGlobalQueryHandlerAO* CGlobalQueryHandlerAO::NewLC(MGlobalQueryObserver& aObserver)
    {
    CGlobalQueryHandlerAO* self = new (ELeave) CGlobalQueryHandlerAO(aObserver);
    CleanupStack::PushL(self);
    self->ConstructL();
    return self;
    }

CGlobalQueryHandlerAO* CGlobalQueryHandlerAO::NewL(MGlobalQueryObserver& aObserver)
    {
    CGlobalQueryHandlerAO* self = CGlobalQueryHandlerAO::NewLC(aObserver);
    CleanupStack::Pop(self);
    return self;
    }

void CGlobalQueryHandlerAO::DoCancel(){}

void CGlobalQueryHandlerAO::StartHandler() { SetActive(); }

void CGlobalQueryHandlerAO::RunL()
    {
    if (iStatus == EAknSoftkeyOk)
        iObserver.ProcessOkOptionL();
    else if(iStatus == EAknSoftkeyCancel)
        iObserver.ProcessCancelOptionL();
    else if(iStatus == EAknSoftkeyDone)
        iObserver.ProcessDoneOptionL();
    else
        iObserver.ProcessSelectedItemL(iStatus.Int());

    Cancel();
    }
```

container / observer

```
#include "GlobalQueryContainer.h" //CGlobalQueryContainer
#include "GlobalQueryHandlerAO.h" //CGlobalQueryHandlerAO
#include <GlobalQuery_0xED0C36BE.rsg> //Resources
#include <StringLoader.h> // StringLoader

void CGlobalQueryContainer::ConstructL(const TRect& aRect)
    {
    CreateWindowL();
    SetRect(aRect);
    ActivateL();

    MakeExampleQueryL();
```


(C) Copyright Nokia 2013. All rights reserved.

```
    }

void CGlobalQueryContainer::MakeExampleQueryL()
    {
    //start the query handler and show global progress dialog
    iQueryHandlerAO = CGlobalQueryHandlerAO::NewL(*this);
    iQueryHandlerAO->StartHandler();

    iGlobalProgressDialog = CAknGlobalProgressDialog::NewL();

    HBufC* progressText = StringLoader::LoadLC(R_GLOBAL_QUERY_PROGRESS_TEXT);

    iCurrentExampleStep = 0;

    iGlobalProgressDialog->ShowProgressDialogL(
                        iQueryHandlerAO->iStatus,
                        *progressText,
                        R_AVKON_SOFTKEYS_CANCEL);

    //start example process and update the progress dialog
    iTimer = CPeriodic::NewL(EPriorityLow);
    iTimer->Start(1000, 1000, TCallBack(TimerCallback, this));

    CleanupStack::PopAndDestroy(); //progressText
    }

CGlobalQueryContainer::~CGlobalQueryContainer()
    {
    //...
    delete iQueryHandlerAO;
    delete iGlobalProgressDialog;

    if(iTimer)
        {
        iTimer->Cancel();
        delete iTimer;
        }
    }

TInt CGlobalQueryContainer::TimerCallback(TAny* aThis)
    {
    static_cast< CGlobalQueryContainer* >(aThis)->GlobalExampleProcessTimerEvent();

    return 0;
    }

void CGlobalQueryContainer::GlobalExampleProcessTimerEvent()
    {
    if (!iGlobalProgressDialog)
        {
        return;
        }

    iCurrentExampleStep++;

    if (iCurrentExampleStep <= KFinalExampleStep)
        {
        iGlobalProgressDialog->UpdateProgressDialog(iCurrentExampleStep, KFinalExampleStep);
        }
    else
        {
        iGlobalProgressDialog->ProcessFinished();
        iTimer->Cancel();
        }
    }

void CGlobalQueryContainer::ProcessOkOptionL()
    {
    //not used
    }
void CGlobalQueryContainer::ProcessCancelOptionL()
    {
    //Cancel pressed, do something...
    iEikonEnv->InfoWinL(_L("Cancel"),_L("Pressed"));
    }
void CGlobalQueryContainer::ProcessDoneOptionL()
    {
    //Progress dialog ready, do something...
    iEikonEnv->InfoWinL(_L("Progress"),_L("Done"));
    }
void CGlobalQueryContainer::ProcessSelectedItemL(const TInt /*aItem*/)
    {
    //not used
    }
```

## Postconditions

The example code shows how to create a global progress dialog, update the progress bar, handle its dismissal, and how the active object handler informs an observer class that the dialog has been closed.

## See also

- Showing a global message query using CAknGlobalMsgQuery
- Showing a global confirmation query using CAknGlobalConfirmationQuery
- Showing a global list query using CAknGlobalListQuery


(C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.