Logs Example

The `CCallLogReader` illustrates how to read all call log entries from the phone's log. The logs are processed from the last to the first entry. The reason for this arrangement is that you could also delete entries while reading them. If deleting while reading from first to last the indexing of the logs could go wrong and cause unexpected error situations.

If you want to monitor the log as it gets updated, you can use CLogMonitor

Link against:

```
LIBRARY logcli.lib
```

Capabilities:

```
CAPABILITY ReadUserData
```

LogReader.cpp

```
#include "LogReader.h"

CCallLogReader* CCallLogReader::NewL(MLogCallBack* aCallBack)
{
    CCallLogReader* self = CCallLogReader::NewLC(aCallBack);
    CleanupStack::Pop(self);
    return self;
}

CCallLogReader* CCallLogReader::NewLC(MLogCallBack* aCallBack)
{
    CCallLogReader* self = new (ELeave) CCallLogReader(aCallBack);
    CleanupStack::PushL(self);
    self->ConstructL();
    return self;
}

CCallLogReader::CCallLogReader(MLogCallBack* aCallBack)
:CActive(CActive::EPriorityStandard),iCallBack(aCallBack)
{
}

CCallLogReader::~CCallLogReader()
{
    Cancel();
    delete iLogView, iLogView = NULL;
    delete iLogFilter, iLogFilter = NULL;
    delete iLogClient, iLogClient = NULL;
    iFsSession.Close();
}

void CCallLogReader::ConstructL(void)
{
    CActiveScheduler::Add(this);

    User::LeaveIfError(iFsSession.Connect());
```


(C) Copyright Nokia 2014. All rights reserved.

```
    iLogClient = CLogClient::NewL(iFsSession);
    iLogView = CLogViewEvent::NewL(*iLogClient);
    iLogFilter = CLogFilter::NewL();

    if(iLogView->SetFilterL(*iLogFilter, iStatus))
    {
        iEngineState = ECreatingView;
        SetActive();
    }
    else
        DoneReadingL(KErrNone);
}

void CCallLogReader::DoCancel()
{
    if(iLogView)
        iLogView->Cancel();

    if(iLogClient)
        iLogClient->Cancel();
}

void CCallLogReader::RunL()
{
    if(iStatus != KErrNone)
        DoneReadingL(iStatus.Int());
    else
        switch (iEngineState)
        {
            case ECreatingView:
               if(iLogView)
               {
                   // The filtered view has been successfully created
                   // so issue a request to start processing logs backwards
                   if(iLogView->LastL(iStatus))
                   {
                       iEngineState = EReadingEntries;
                       SetActive();
                   }
                   else
                       DoneReadingL(KErrNone);
               }
               break;

            case EReadingLast:
               if(iLogView)
               {
                   iLogView->FirstL(iStatus);
                   iEngineState = EReadingEntries;
                   SetActive();
               }
               break;

            case EReadingEntries:
               if(iLogView)
               {
```


(C) Copyright Nokia 2014. All rights reserved.

```
                    // since we are working from last-to-first
                    // you could also delete entries at this point if necessary
                    iCallBack->HandleLogEventL(iLogView->Event());

                    iEngineState = EReadingEntries;
                    if(iLogView->PreviousL(iStatus))
                        SetActive();
                    else
                        DoneReadingL(KErrNone);
                }
                break;

            default:
               break;
        }
}


void CCallLogReader::DoneReadingL(TInt aError)
{
    iCallBack->LogProcessed(aError);
}
```

LogReader.h

```
#include <F32FILE.H>
#include <LOGVIEW.H>
#include <logcli.h>

class MLogCallBack
{
    public:
        virtual void HandleLogEventL(const CLogEvent& event) = 0;
        virtual void LogProcessed(TInt aError) = 0;
};


class CCallLogReader: public CActive
{
    enum TCallLogReaderState
    {
        ECreatingView,
        EReadingEntries,
        EReadingLast
    };

    public: // constructors and destructor
        static CCallLogReader* NewL(MLogCallBack* aCallBack);
        static CCallLogReader* NewLC(MLogCallBack* aCallBack);
        ~CCallLogReader();

    public: // from CActive
        void DoCancel();
        void RunL();
```


(C) Copyright Nokia 2014. All rights reserved.

```
    private: // constructors
        CCallLogReader(MLogCallBack* aCallBack);
        void ConstructL();
        void DoneReadingL(TInt aError);

    private: // data
        TCallLogReaderState iEngineState;
        CLogClient*         iLogClient;
        CLogViewEvent*      iLogView;
        CLogFilter*         iLogFilter;
        MLogCallBack*       iCallBack;
        RFs                 iFsSession;
};
```

Related Links

- Logs monitoring Example

(C) Copyright Nokia 2014. All rights reserved.