Pausing a Symbian C++ app on an incoming SMS

Overview

Sometimes the application (for example, a game) needs to be paused when a message is received. This snippet demonstrates how to observe incoming messages, thus enabling acting on them.

This snippet can be self-signed.

MMP file

The following libraries are required:

```
LIBRARY msgs.lib
```

Header file: CSMSEngine.h

```
#ifndef __CSMSENGINE_H_
#define __CSMSENGINE_H_

#include <e32base.h>    // CBase
#include <msvapi.h>     // MMsvSessionObserver

class MSMSEngineObserver
    {
    public:
        virtual void SMSReceived() = 0;
    };

class CSMSEngine : public CBase, public MMsvSessionObserver
    {
    public:
        /**
         * 2nd phase constructor.
         * @param aObserver the observer which gets notified when message
         * events occur
         * @return A pointer to the created instance of CSMSEngine
         */
        static CSMSEngine* NewL(MSMSEngineObserver* aObserver);

        /**
         * 2nd phase constructor.
         * @param aObserver the observer which gets notified when message
         * events occur
         * @return A pointer to the created instance of CSMSEngine
         */
        static CSMSEngine* NewLC(MSMSEngineObserver* aObserver);

        /**
         * Destructor.
         */
        ~CSMSEngine();

    public:
        /**
         * From MMsvSessionObserver.
         * Called when a session event occurs.
         */
        void HandleSessionEventL(TMsvSessionEvent aEvent,
                TAny* aArg1,
                TAny* aArg2,
                TAny* aArg3);

    private:
        /**
         * Symbian OS default constructor.
         * @param aObserver the observer which gets notified when message
         * events occur
         */
        CSMSEngine(MSMSEngineObserver* aObserver);

        /**
         * 2nd phase constructor.
         * Initializes objects.
         */
        void ConstructL();

    private:
        // Observes the SMS engine. Gets notified when message events occur.
        MSMSEngineObserver* iObserver;

        // Session with the messaging server
        CMsvSession* iMsvSession;
    };

#endif /*__CSMSENGINE_H_*/
```

Source file: CSMSEngine.cpp

(C) Copyright Nokia 2013. All rights reserved.

```
#include <msvids.h> // Folder Ids

#include "CSMSEngine.h"

const TMsvId KDrafts = KMsvDraftEntryId;
const TMsvId KInbox = KMsvGlobalInBoxIndexEntryId;

// When using the emulator the observed folder is Drafts, otherwise Inbox.
#ifdef __WINS__
    const TMsvId KObservedFolderId = KDrafts;
#else
    const TMsvId KObservedFolderId = KInbox;
#endif

/**
 * 2nd phase constructor.
 */
CSMSEngine* CSMSEngine::NewL(MSMSEngineObserver* aObserver)
    {
    CSMSEngine* self = CSMSEngine::NewLC(aObserver);
    CleanupStack::Pop(self);
    return self;
    }

/**
 * 2nd phase constructor.
 */
CSMSEngine* CSMSEngine::NewLC(MSMSEngineObserver* aObserver)
    {
    CSMSEngine* self = new (ELeave) CSMSEngine(aObserver);
    CleanupStack::PushL(self);
    self->ConstructL();
    return self;
    }

/**
 * Symbian OS default constructor.
 */
CSMSEngine::CSMSEngine(MSMSEngineObserver* aObserver) : iObserver(aObserver)
    {
    }

/**
 * 2nd phase constructor.
 */
void CSMSEngine::ConstructL()
    {
    // SMS automatic receiving needs a session to the messaging server
    iMsvSession = CMsvSession::OpenAsyncL(*this);
    }

/**
 * Destructor.
 */
CSMSEngine::~CSMSEngine()
    {
    if (iMsvSession)
        {
        iMsvSession->Cancel();
        }
    delete iMsvSession;
    iMsvSession = NULL;
    }

/**
 * From MMsvSessionObserver.
 * Called when a session event occurs.
 */
void CSMSEngine::HandleSessionEventL(TMsvSessionEvent aEvent, TAny* aArg1,
        TAny* aArg2, TAny* /*aArg3*/)
    {
    switch (aEvent)
        {
        case EMsvEntriesChanged:
            {
            // If the event happens in the observed folder, notify the observer
            if (aArg2 && *(static_cast<TMsvId*>(aArg2)) == KObservedFolderId)
                {
                iObserver->SMSReceived();
                }
            break;
            }
        }
    }
```

Header file: CAppUi.h

```
#ifndef __CAPPUI_H__
#define __CAPPUI_H__

#include "CSMSEngine.h"

// The UI class observes the SMS engine
class CAppUi : public MSMSEngineObserver
    {
    public:
        /**
         * 2nd phase constructor.
         * Initializes objects.
         */
        void ConstructL();

        // ...

    private: // Functions from base classes
        /**
         * From MSMSEngineObserver
         */
        void SMSReceived();

    private: // Data
        CSMSEngine* iSMSEngine;
    };

#endif // __CAPPUI_H__
```


(C) Copyright Nokia 2013. All rights reserved.

Source file: CAppUi.cpp

```
#include "CSMSEngine.h"
```

```
/**
 * 2nd phase constructor.
 */
void CAppUi::ConstructL()
    {
    // ...

    // Create the SMS engine and set this class as its observer
    iSMSEngine = CSMSEngine::NewL(this);

    // ...
    }

void CAppUi::SMSReceived()
    {
    // A message is received. Pause the application.
    // ...
    }
```

Postconditions

When a message is received, the observer's (here `CAppUi`'s) `SMSReceived` is called.

See also

- Messaging: Nokia Developer Wiki - Messaging
- SMS operations using Symbian C++: Nokia Developer Wiki - SMS Operations

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.