

This article intends to explain how to handle most of the events on a call using CTelephony in 3rd Edition

Events handled here are:

1. Call Ringing
2. Call Answered
3. Call connecting
4. Call connected
5. Call disconnected
6. Dial call

The implementation is so simple. The idea is simply active object listening to change in voice line

Contents

- [1 Headers Required](#)
- [2 Libraries Required](#)
- [3 Capability Required](#)
- [4 Code](#)
- [5](#)
[MPhoneReceiverObserver](#)
- [6 CPhoneReceiver](#)
[Declaration](#)
- [7 CPhoneReceiver](#)
[Implementation](#)
- [8 Output](#)

Headers Required

```
#include <Etel3rdParty.h>
```

Libraries Required

```
LIBRARY etel3rdparty.lib
```

Capability Required

```
Capability NetworkServices
```

Code

The following code assumes that you have a class for log text (CMyLog).

Define observer interface to be implemented by class you want it to be notified with events (if you like)

MPhoneReceiverObserver

```
#ifndef MPHONERECEIVEROBSERVER_H
#define MPHONERECEIVEROBSERVER_H

class MPhoneReceiverObserver
{
public:
virtual void CallRinging() {}

virtual void CallAnswered() {}

virtual void DialingCall() {}

virtual void CallConnecting() {}

virtual void CallConnected() {}

virtual void CallDisConnected() {}

};

#endif // MPHONERECEIVEROBSERVER_H
```

Define the active object that will listen to any change in voice line status

CPhoneReceiver Declaration

```
#include <e32base.h> // For CActive, link against: euser.lib
#include "Etel3rdParty.h" //For CTelephony, link against etel3rdparty.lib

#include "MPhoneReceiverObserver.h"
#include "MyLog.h"

class CPhoneReceiver : public CActive
{
public:
// C++ constructor
CPhoneReceiver(MPhoneReceiverObserver& aPhoneReceiverObserver);

// Second-phase constructor
void ConstructL();

// Cancel and destroy
~CPhoneReceiver();

public: // New functions
// Function for making the initial request
void StartL();

private: // From CActive
```

How_to_handle_call_events_using_CTelephony_3rd_Edition

```
// Handle completion
void RunL();

// How to cancel me
void DoCancel();

private:
//Define object to CTelephony to manage calls
    CTelephony      ;          iTelephony

    CTelephonyInfoV1      ;iCurrentCallInfo
    CTelephonyInfoV1Pckg  ;iCurrentStatusPckg

    CTelephony          ;          iCallID

    MPhoneReceiverObserver      ;iPhoneReceiverObserver

};

#endif
```

CPhoneReceiver Implementation

```
#include "PhoneReceiver.h"

CPhoneReceiver::CPhoneReceiver(MPhoneReceiverObserver& aPhoneReceiverObserver)
:    (CPriorityStandard),
    iCurrentStatusPckg(),
    iPhoneReceiverObserver(aPhoneReceiverObserver)
{
}

void CPhoneReceiver::ConstructL()
{
//Create new object to telephony
    iTelephony::NewL();
    CActiveSchedulerAdd(this); // Add to scheduler
}

CPhoneReceiver::~CPhoneReceiver()
{
    ()Cancel any request, if outstanding

delete iTelephony;
}

void CPhoneReceiver::DoCancel()
{
    iTelephony->Async(CTelephony::EVoiceLineStatusChangeCancel);
}

void CPhoneReceiver::StartL()
{
    ()Cancel any request, just to be sure

//Notify of change in telephone line
```

How_to_handle_call_events_using_CTelephony_3rd_Edition

```
    iTelephonyChange( iStatus, CTelephony::EVoiceLineStatusChange,
        iCurrentStatusPckg

    SetActive scheduler a request is active
}

void CPhoneReceiver::RunL()
{
    if(iStatus.Int() == KErrNone)
    {
        //Get call status
        :TCalCTelephonyAllStatus = iCurrentStatusPckg().iStatus;

    switch(callStatus)
    {
    case CTelephony::EStatusRinging:
        CáPhónBảngngivơObserver.
    break;

    case CTelephony::EStatusAnswering:
        CáPhónReceivơObserver.
    break;

    case CTelephony::EStatusDialling:
        DiHỏngReceivơObserver.
    break;

    case CTelephony::EStatusConnecting:
        CáPhónReceivơObserver.
    break;

    case CTelephony::EStatusConnected:
        CáPhónReceivơObserver.
    break;

    case CTelephony::EStatusDisconnecting:
        CáPhónReceivơObserver.
    break;

    default:
#ifdef __USE_PHONE_RECEIVER_LOG__
        <50> num;    TBuf
        Copy(_L("Default: ") num.
        AppendNum(callStatus) num.
        ::OpenAndWriteToFile(MyFile(num);
#endif
    break;
    }
}
else
{
#ifdef __USE_PHONE_RECEIVER_LOG__
    <50> num;    TBuf
    AppendNum(iStatus.Int());
    Append(_L(" num!"));
    :TCalCTelephonyAllStatus = iCurrentStatusPckg().iStatus;
    AppendNum(callStatus);
    ::OpenAndWriteToFile(MyFile(num);
#endif
}

//Notify every time before activate object
```

How_to_handle_call_events_using_CTelephony_3rd_Edition

```
CTelephonyChange( iStatus, CTelephony::EVoiceLineStatusChange,  
iCurrentStatusPckg  
  
if(!IsActive())  
    ();    SetActive  
  
}
```

Output

The following is a sample for the log file written above

Log:25-4-2009-15-18-53

////////////////////

Call Ringing

////////////////////

Call Answered

////////////////////

Call connected

////////////////////

Call disconnected

////////////////////

Default: 1 //Line Idle

////////////////////