

This code example will help you in getting different Network Parameters like: Cell Id, Network Id, Country ID and Network Name.

"Danger, Will Robinson!"

Please note that ETel3rdParty (CTelephony) is an API designed to be used asynchronously and that any attempt to do otherwise risks to either panic or freeze your application. In particular, an application using the code presented here will become irresponsive when the phone does not have network connectivity (temporary signal drop or Offline mode caused by profile change or missing SIM)

Contents

- [1](#)
[NetworkApp.h](#)
- [2](#)
[NetworkApp.cpp](#)
- [3](#)
[Instructions](#)
- [4](#) [Related](#)
[Links](#)

NetworkApp.h

```
#include <e32base.h>
#include <Etel3rdParty.h>

class CNetworkApp : public CActive
{
private:
    void ConstructL();
    CTelephony* iTelephony;
    CTelephony::TNetworkInfoV1 iNetworkInfoV1;
    CTelephony::TNetworkInfoV1Pckg iNetworkInfoV1Pckg;

public:
    CNetworkApp(
        TUint& CellId, TDes& NetworkId, TDes& CountryId, TDes& LongName);
    static void GetNetworkParameters(
        TUint& CellId, TDes& NetworkId, TDes& CountryId, TDes& LongName);
    ~CNetworkApp();
    TUint& iCellID;
    TDes& iNetworkID;
    TDes& iCountryCODE;
    TDes& iLongNAME;

private:
    /*
     * These are the pure virtual methods from CActive that
     * MUST be implemented by all active objects
     */
    void RunL();
};
```

```
void DoCancel();
};
```

NetworkApp.cpp

```
#include "NetworkApp.h"

void CNetworkApp::GetNetworkParameters (
    TUInt& aCellID, TDes& aNetworkID, TDes& aCountryCODE,
    TDes& aLongName)
{
    CNetworkApp* self=
        new(ELeave) CNetworkApp(
            aCellID, aNetworkID, aCountryCODE, aLongName);
    CleanupStack::PushL(self);
    self->ConstructL();
    CleanupStack::PopAndDestroy(self);
}

void CNetworkApp::ConstructL()
{
    iTelephony = CTelephony::NewL();
    CActiveScheduler::Add(this);

    iTelephony->GetCurrentNetworkInfo(iStatus, iNetworkInfoV1Pckg);
    SetActive();
    CActiveScheduler::Start();
}

CNetworkApp::CNetworkApp(
    TUInt& aCellID, TDes& aNetworkID, TDes& aCountryCODE, TDes& aLongName):
    CActive(EPriorityStandard), iNetworkInfoV1Pckg(iNetworkInfoV1),
    iCellID(aCellID), iNetworkID(aNetworkID), iCountryCODE(aCountryCODE),
    iLongNAME(aLongName)
{
    //default constructor
}

void CNetworkApp::RunL()
{
    if(iStatus==KErrNone)
    {
        iCellID = iNetworkInfoV1.iCellId;
        iNetworkID = iNetworkInfoV1.iNetworkId;
        iCountryCODE = iNetworkInfoV1.iCountryCode;
        iLongNAME= iNetworkInfoV1.iLongName;
        CActiveScheduler::Stop();
    }
}

void CNetworkApp::DoCancel()
{
    iTelephony->CancelAsync(CTelephony::EGetCurrentNetworkInfoCancel);
}

CNetworkApp::~CNetworkApp()
{
    if(iTelephony)
    {
```

```
Cancel();  
delete iTelephony;  
iTelephony = NULL;  
}  
}
```

Instructions

Now do the following steps:

- 1) Create a sample "Hello World" type application.
- 2) Put **NetworkApp.h** file in your application's "inc" folder.
- 3) Put **NetworkApp.cpp** file in your application's "src" folder.
- 4) Open .mmp file and add entry **SOURCE NetworkApp.cpp**.
- 5) Open .mmp file and add entry **LIBRARY etel3rdparty.lib**.
- 7) Now you need to include following header in your class to read Network:
#include "NetworkApp.h". Let's say in for e.g.: **"CYrAppUi.h"**
- 8) You can then call the static function from any of your Commands.
For ex: in your **CYrAppUi.h** you need to include **"NetworkApp.h"**
And Call it like:

```
case EReadNetworkCommand1:  
{  
    TInt CellId;  
    TBuf<30> NetworkId;  
    TBuf<30> CountryId;  
    TBuf<30> OperatorLongName;  
    CNetworkApp::GetNetworkParameters(  
        CellId, NetworkId, CountryId, OperatorLongName );  
  
    TBuf<160> iDisplayString;  
    iDisplayString.Format(  
        _L( "CellId-%d\nNetworkID-%S\nCountryID-%S\nNETWORK-%S" ),  
        CellId, &NetworkId, &CountryId, &OperatorLongName );  
  
    CAknInformationNote* informationNote = new ( ELeave )  
        CAknInformationNote;  
    informationNote->ExecuteLD(iDisplayString);  
}
```

- 9) Open .mmp file and add entry **CAPABILITY ReadDeviceData**.
- 10) You need to sign resulted **.Sis** file with your developer certificate to get it installed on your phone.

You can find a working application which can be built with [Carbide.c++](#) Here it goes: [Network Information.zip](#)
[Here](#) is a library with similar functionality but for [S60](#) 2nd edition SDKs.

Related Links

- [DevInfo - Get the IMEI, IMSI, CellId etc., synchronously on 3.x devices.](#)
-