Make call with CTelephony

`CCallDialer` example illustrates how to dial a new call using `CTelephony`. Note that this example is intended to be used only with S60 3rd Edition devices, and does not work with pre-3rd Edition devices.

To use this example first implement the callback interface function in the implementing class and then construct an instance of the `CCallDialer`. The call dialing starts automatically after the construction. In case the process is successful and the callee answers the call, `RunL()` will be called with `KErrNone`; otherwise the `aError` argument variable will indicate the error.

Headers required:

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

Library required:

```
LIBRARY etel3rdparty.lib
```

Capabilities required:

```
CAPABILITY NetworkServices
```

To instantiate the CCallDialer, from the observing class, use something like this:

```
iCallDialer = CCallDialer::NewL(*this, phoneNumber1);
```

Then, in the notifier method of the instantiating class, you may remove the dialer and optionally check status:

```
void CBoopCtlContainer::CallDialedL(TInt aError)
{
    delete iCallDialer, iCallDialer = NULL;
}
```

You may subsequently dial another number by going try the same sequence:

```
iCallDialer = CCallDialer::NewL(*this, phoneNumber2);
```

MakeCall.cpp

```
#include "MakeCall.h"

CCallDialer* CCallDialer::NewL(MDialObserver& aCallBack, const TDesC& aNumber)
{
    CCallDialer* self = CCallDialer::NewLC(aCallBack, aNumber);
    CleanupStack::Pop(self);
    return self;
}

CCallDialer* CCallDialer::NewLC(MDialObserver& aCallBack, const TDesC& aNumber)
{
    CCallDialer* self = new (ELeave) CCallDialer(aCallBack);
```


(C) Copyright Nokia 2014. All rights reserved.

```
    CleanupStack::PushL(self);
    self->ConstructL(aNumber);
    return self;
}


CCallDialer::~CCallDialer()
{
    Cancel();
    delete iTelephony;
}

void CCallDialer::ConstructL(const TDesC& aNumber)
{
    iTelephony = CTelephony::NewL();
    CTelephony::TTelNumber telNumber(aNumber);

    iCallParams.iIdRestrict = CTelephony::ESendMyId;

    iTelephony->DialNewCall(iStatus, iCallParamsPckg, telNumber, iCallId);
    SetActive();
}

CCallDialer::CCallDialer(MDialObserver& aObserver)
: CActive(EPriorityNormal),iObserver(aObserver), iCallParamsPckg(iCallParams)
{
    CActiveScheduler::Add(this);
}

void CCallDialer::RunL()
{
    iObserver.CallDialedL(iStatus.Int());
}

void CCallDialer::DoCancel()
{
    iTelephony->CancelAsync(CTelephony::EDialNewCallCancel);
}
```

MakeCall.h

```
#include <Etel3rdParty.h>
// link to etel3rdparty.lib

class MDialObserver
{
    public:
        virtual void CallDialedL(TInt aError) = 0;
};

class CCallDialer : public CActive
{
    public:
        static CCallDialer* NewL(MDialObserver& aCallBack, const TDesC& aNumber);
        static CCallDialer* NewLC(MDialObserver& aCallBack, const TDesC& aNumber);
```


(C) Copyright Nokia 2014. All rights reserved.

```
        ~CCallDialer();

    protected:
        CCallDialer(MDialObserver& aObserver);
        void ConstructL(const TDesC& aNumber);

    private:
        void RunL();
        void DoCancel();

    private:
        MDialObserver&                iObserver;
        CTelephony*                   iTelephony;
        CTelephony::TCallId           iCallId;
        CTelephony::TCallParamsV1     iCallParams;
        CTelephony::TCallParamsV1Pckg iCallParamsPckg;
};
```


(C) Copyright Nokia 2014. All rights reserved.