



CSMSReceiver example illustrates how you can wait incoming SMS message and extract information from them as soon as they arrive into the inbox. After constructing the iSession inside the ConstructL()-function, the HandleSessionEventL()-function will be called by the OS to notify your class on CMsvSession events.

As shown in the example, inside this function only EMsvEntriesCreated and EMsvEntriesMoved are checked which after the code checks if the event happened for the inbox, which after the HandleEntryL()-function is used to handle individual SMS messages. You could use all functions defined for the CBaseMtm to extract information of the message (and you could also cast the CBaseMtm to CSmsClientMtm for SMS specific functionality), with this example only the body of the message is forwarded to the callback class.

## SMS\_Receiver.cpp

```

CSMSReceiver* CSMSReceiver::NewL(MSMSRecCallBack& aObserver)
{
    CSMSReceiver new(ELeave)CSMSReceiver(aObserver);
    ->ConstructL();
    return self;
}

CSMSReceiver::CSMSReceiver(MSMSRecCallBack& aObserver):iObserver(aObserver)
{
}

CSMSReceiver::~CSMSReceiver()
{
    delete iMtmRegistry;
    iMtmRegistry = NULL;

    delete iSession;
    iSession = NULL;
}

void CSMSReceiver::ConstructL(void)
{
    iSession = CSMSvSession::OpenSyncL(*this);
    iMtmRegistry = CSmsClientMtmRegistry::NewL(*iSession);
}

void CSMSReceiver::HandleSessionEventL(TMsvSessionEvent aEvent, TAny* aArg1, TAny* aArg2, TAny* /
{
    switch (aEvent)
    {
        case EMsvEntriesChanged:
        {
        }
        break;
        case EMsvEntriesCreated:
        case EMsvEntriesMoved:
        {
            * entryId;
            entryId_c = cast<TMsvId*>(aArg2); // entry id from the session event

            if ( *entryId == KMsvGlobalInBoxIndexEntryId ) // new entry has been created in Inb
        {

```

## SMS\_Receiver

```

        TMsVSelectionOrdering sort
        SetShowInvisibleEntries(EFalse); // we dont want to handle the invisible entries
// Take a handle to the Inbox entry
        * parentEntry = CMsvEntry::NewL(CMsvSession, KMsvGlobalInBoxIndexEntryId, sort);
        ::PushL(parentEntry); CleanupStack

        * entries = static_cast<CMsvEntrySelection*>(aArg1);
if(entries)
{
    //Process each created entry, one at a time.
    for(TInt i = 0; i < entries->Count(); i++)
    {
        (entry->HandleEntryL);
    }
}

        ::PopAndDestroyL(parentEntry)
    }
}
break;
case EMsvCloseSession:
    iSession->CloseMessageServer();
    break;
default:
    // All other events are ignored
    break;
}
}

void CSMSReceiver::HandleEntryL(TMsvId& aEntId)
{
    CMsvEntry* entry = iSession->GetEntryL(aEntId);
    CleanupStack(entry);

if(entry->Entry().iMtm == KUidMsgTypeSMS)
{
    * SmsMtm = CSmsMtmRegistry->NewMtmL(KUidMsgTypeSMS);
if(SmsMtm)
{
    ::Pop(1); //entry cleanupStack
    ::PushL(SmsMtm); //entry cleanupStack
    entry->SetCurrentEntryL(SmsMtm);
    entry->LoadMessageL(); SmsMtm

    * BodyBuffer = HBufC::NewLC(SmsMtm->Body().DocumentLength());
        (BodyBuffer->Data().Data(), BodyBuffer->Data().Data());

    entry->Body().Extract(BodyBuffer, 0, SmsMtm->Body().DocumentLength());

    GotSMSMessageL(BodyBuffer);

    ::PopAndDestroyL(SmsMtm, BodyBuffer)
}
else
{
    ::PopAndDestroyL(entry)
}
else
{
    ::PopAndDestroyL(1); //entry
}
}

```

}

## SMS\_Receiver.h

```

class MSMSRecCallBack
{
public:
virtual void GotSMSMessageL(const TDesC& aMessage) = 0;
};

class CSMSReceiver : public CBase, MMsvSessionObserver
{
public:
static CSMSReceiver* NewL(MSMSRecCallBack& aObserver);
~CSMSReceiver() {}
protected:
CSMSReceiver(MSMSRecCallBack& aObserver);
void ConstructL(void);
void HandleSessionEventL(TMsvSessionEvent aEvent, TAny *aArg1, TAny *aArg2, TAny *aArg3);
private:
void HandleEntryL(TMsvId& aEntId);
private:
MSMSRecCallBack iObserver
CMsvSession ; iSession
CClientMtmRegistry iMtmRegistry
};

```

If you are looking for a full example application for this functionality, you could have a look into an example for monitoring and deleting log entries for SMS messages, especially for Delivery reports is also illustrated in [SMS DeliveryReport Deleting Example](#).

## Related Links:

- [Sending SMS with RSendAs](#)
- [SMS Utilities API](#)
- [Reading SMS from Inbox](#)
- [Sending SMS in S60 3rd Edition - MTM](#)
- [SMS Operations](#)
- [How to send an SMS using sockets](#)
- [Sending-Receiving SMS through an Exe \(Server\)](#)
- [Create Local SMS](#)
- [How to Open SMS or MMS Editor](#)