



## Creating the GPRS Internet Access Point (IAP) OS version independent

In order to create the IAP, CCommsDatabase is used. The following code example creates all the table entries manually. It actually duplicates the emulator's "Winsock" IAP initial settings. CreateNewIAP () is the main function.

```

#include <commdbconnpref.h>
#include <commdb.h>
#include <cdbpreftable.h>

TUInt32 CreateWapApL(CCommsDatabase* db)
{
    (KWapApName, "MyAP WAP");
    TUInt32 WapId
    CCommsDbTable* view db->OpenTableLC(TPtrC(WAP_ACCESS_POINT));
    ::LeaveIfError (view->InsertRecord(WapId));
    ->WriteTextL(TPtrC(COMMDB_NAME), KWapApName);
    ->WriteTextL(TPtrC(WAP_CURRENT_BEARER), TPtrC(WAP_IP_BEARER));

    ::LeaveIfError (view->PutRecordChanges());
    delete view;

    return WapId;
}

TUInt32 CreateGprsServiceL(CCommsDatabase* db)
{
    (KGprsName, "MyAP Gprs");
    (KSettingAPN, "Winsock");

    TUInt32 gprsId
    CCommsDbTable* view db->OpenTableLC(TPtrC(OUTGOING_GPRS));
    ::LeaveIfError (view->InsertRecord(gprsId));

    ->WriteTextL(TPtrC(COMMDB_NAME), KGprsName);
    ->WriteTextL(TPtrC(GPRS_APN), KSettingAPN);
    ->WriteIntL(TPtrC(GPRS_PDP_TYPE), 0);
    view ->WriteBoolL(TPtrC(GPRS_IP_ADDR_FROM_SERVER), EFalse);
    view ->WriteBoolL(TPtrC(GPRS_IP_DNS_ADDR_FROM_SERVER), EFalse);
    view ->WriteBoolL(TPtrC(GPRS_IP6_DNS_ADDR_FROM_SERVER), EFalse);
    view ->WriteTextL(TPtrC(GPRS_IF_AUTH_NAME), _L(""));
    view ->WriteTextL(TPtrC(GPRS_IF_AUTH_PASS), _L(""));
    view ->WriteBoolL(TPtrC(GPRS_IF_PROMPT_FOR_AUTH), EFalse);
    view ->WriteUIntL(TPtrC(SERVICE_IF_AUTH_RETRIES), 0);
    view ->WriteTextL(TPtrC(GPRS_IF_NETWORKS), _L("tcp,udp,icmp,icmp6"));
    view ->WriteBoolL(TPtrC(GPRS_HEADER_COMPRESSION), EFalse);
    view ->WriteBoolL(TPtrC(GPRS_DATA_COMPRESSION), EFalse);
    view ->WriteUIntL(TPtrC(GPRS_REQ_PRECEDENCE), 0);
    view ->WriteUIntL(TPtrC(GPRS_REQ_DELAY), 0);
    view ->WriteUIntL(TPtrC(GPRS_REQ_RELIABILITY), 0);
    view ->WriteUIntL(TPtrC(GPRS_REQ_PEAK_THROUGHPUT), 0);
    view ->WriteUIntL(TPtrC(GPRS_REQ_MEAN_THROUGHPUT), 0);
    view ->WriteUIntL(TPtrC(GPRS_MIN_PRECEDENCE), 0);
    view ->WriteUIntL(TPtrC(GPRS_MIN_DELAY), 0);

```

## Create\_new\_internet\_access\_point

```
view      ->WriteUIntL(TPtrC(GPRS_MIN_RELIABILITY), 0);
view      ->WriteUIntL(TPtrC(GPRS_MIN_PEAK_THROUGHPUT), 0);
view      ->WriteUIntL(TPtrC(GPRS_MIN_MEAN_THROUGHPUT), 0);
view      ->WriteBoolL(TPtrC(GPRS_ANONYMOUS_ACCESS), EFalse);
view      ->WriteBoolL(TPtrC(GPRS_ENABLE_LCP_EXTENSIONS), EFalse);
view      ->WriteBoolL(TPtrC(GPRS_DISABLE_PLAIN_TEXT_AUTH), EFalse);
view      ->WriteUIntL(TPtrC(GPRS_AP_TYPE), 2);
view      ->WriteUIntL(TPtrC(GPRS_QOS_WARNING_TIMEOUT), -1);

    ::LeaveIfError(view->PutRecordChanges());
delete view;

return gprsId;
}

TUInt32 CreateLocationL(CCommsDatabase* db)
{
    (KLocName, "MyAP Location");
    TUInt32 LocId
    CCommsDbTable* view db->OpenTableLC(TPtrC(LOCATION));
    ::LeaveIfError(view->InsertRecord(LocId));

    ->WriteTextL(TPtrC(COMMDB_NAME), KLocName);
    ->WriteTextL(TPtrC(LOCATION_INTL_PREFIX_CODE), _L("+"));
    ->WriteTextL(TPtrC(LOCATION_NAT_PREFIX_CODE), _L("0"));
    ->WriteTextL(TPtrC(LOCATION_NAT_CODE), _L("44"));
    ->WriteIntL(TPtrC(LOCATION_PAUSE_AFTER_DIAL_OUT), 0);
    ->WriteBoolL(TPtrC(LOCATION_MOBILE), ETrue);
    ->WriteBoolL(TPtrC(LOCATION_USE_PULSE_DIAL), EFalse);
    ->WriteBoolL(TPtrC(LOCATION_WAIT_FOR_DIAL_TONE), EFalse);

    ::LeaveIfError(view->PutRecordChanges());
delete view;

return LocId;
}

TUInt32 CreateBearerL(CCommsDatabase* db)
{
    (KAgentName, "wsocka.agt");
    (KIfName, "wsocki");
    (KLocName, "foo");
    (KBearerName, "MyAP Bearer");

    TUInt32 BearerId
    CCommsDbTable* view db->OpenTableLC(TPtrC(LAN_BEARER));
    ::LeaveIfError(view->InsertRecord(BearerId));

    ->WriteTextL(TPtrC(COMMDB_NAME), KBearerName);
    ->WriteTextL(TPtrC(IF_NAME), KIfName);
    ->WriteTextL(TPtrC(LAN_BEARER_LDD_NAME), KIfName);
    ->WriteTextL(TPtrC(LAN_BEARER_PDD_NAME), KIfName);
    ->WriteTextL(TPtrC(AGENT_NAME), KAgentName);

    ->WriteIntL(TPtrC(LAST_SOCKET_ACTIVITY_TIMEOUT), -1);
    ->WriteIntL(TPtrC(LAST_SESSION_CLOSED_TIMEOUT), -1);
    ->WriteIntL(TPtrC(LAST_SOCKET_CLOSED_TIMEOUT), -1);

    ::LeaveIfError(view->PutRecordChanges());
delete view;

return BearerId;
}
```

## Create\_new\_internet\_access\_point

```
}

TUint32 CreateNetworkL(CCommsDatabase* db)
{
    (KNetName, "MyAP Network");
    TUint32 NetId
    CCommsDbTableView view(db->OpenTableLC(TPtrC(NETWORK)));
    ::LeaveIfError(view->InsertRecord(NetId));

    ->WriteTextL(TPtrC(COMMDB_NAME), KNetName);

    ::LeaveIfError(view->PutRecordChanges());
    delete view;

    return NetId;
}

void CreateConnectionPreferencesL(CCommsDatabase* db, TUint32 Iap)
{
    CCommsDbConnectionPrefTableView=
    ->OpenConnectionPrefTableLC(ECommDbConnectionDirectionOutgoing);

    CCommsDbConnectionPrefTableView.IapBearer bearerinfo;
    bearerinfo.Set(-1;
    bearerinfo.Iap;

    CCommsDbConnectionPrefTableView.IapConnectionPref pref;
    iRapkg=0;
    iDirPref=ECommDbConnectionDirectionOutgoing;
    iDirPref =ECommDbDialogPrefDoNotPrompt;
    iBepref=bearerinfo;

    ->InsertConnectionPreferenceL(pref);
    delete view;
}

TUint32 CreateIAPL(CCommsDatabase* db, TUint32 Bearer, TUint32 Network,
                  TUint32 Service, TUint32 Location)
{
    (KIAPName, "MyAP");
    TUint32 IapId
    CCommsDbTableView view(db->OpenTableLC(TPtrC(IAP)));
    ::LeaveIfError(view->InsertRecord(IapId));

    ->WriteTextL(TPtrC(COMMDB_NAME), KIAPName);
    ->WriteTextL(TPtrC(IAP_SERVICE_TYPE), TPtrC(OUTGOING_GPRS));
    ->WriteTextL(TPtrC(IAP_BEARER_TYPE), TPtrC(LAN_BEARER));
    ->WriteIntL(TPtrC(IAP_BEARER), Bearer);
    ->WriteIntL(TPtrC(IAP_NETWORK), Network);
    ->WriteIntL(TPtrC(IAP_SERVICE), Service);
    ->WriteIntL(TPtrC(LOCATION), Location);
    ->WriteIntL(TPtrC(IAP_NETWORK_WEIGHTING), 0);

    ::LeaveIfError(view->PutRecordChanges());
    delete view;

    return IapId;
}
```

## Create\_new\_internet\_access\_point

```
void CreateWapIPBearerL(CCommsDatabase* db, TUint32 Iap, TUint32 WapIap)
{
    (KWapBearer, "MyAP Wap Bearer");
    TUint32 Id
    CCommsDatabase::View db->OpenTableLC(TPtrC(WAP_IP_BEARER));
    ::LeaveIfError(view->InsertRecord(Id));

    ->WriteTextL(TPtrC(COMMDB_NAME), KWapBearer);
    ->WriteIntL(TPtrC(IAP), Iap);
    ->WriteIntL(TPtrC(WAP_ACCESS_POINT_ID), WapIap);
    ->WriteIntL(TPtrC(WAP_WSP_OPTION), EWapWspOptionConnectionless);
    ->WriteBoolL(TPtrC(WAP_SECURITY), EFalse); \
    ->WriteTextL(TPtrC(WAP_GATEWAY_ADDRESS), _L("0.0.0.0"));
    ->WriteIntL(TPtrC(WAP_PROXY_PORT), 0);

    ::LeaveIfError(view->PutRecordChanges());
    delete view;
}

void CreateNewIAP ()
{
    CCommsDatabase* db = CCommsDatabase::NewL();

    TUint32 CreateBearerL(db);
    TUint32 CreateNetworkL(db);
    TUint32 CreateSprsServiceL(db);
    TUint32 CreateLocationL(db);
    TUint32 CreateIAPL(db, Bearer, Network, Service, Location);

    CreateConnectionPref(db, IAP);

    TUint32 CreateWapApL(db);
    CreateWapIPBearerL(db, IAP, WapAp);

    delete db;
}
```

Need to link with `commdb.lib` in the `.mmp` file.

```
LIBRARY commdb.lib
```

The code requires `WriteDeviceData` capability to run on devices based on Symbian OS v9. So add them in `mmp` file.

```
Capability WriteDeviceData
```

## Related Link

- [IAP](#)