How to export landmarks from database to file

Class **CPosLandmarkEncoder** allows to encode landmark content. You can use this class for exporting landmarks from database to local files. The type (e.g. the mime type) must be specified for creating instance of this class. You can use following mime type for exporting landmarks to the XML-file: "**application/vnd.nokia.landmarkcollection+xml**".

Next code snippet demonstrates how to export all local landmark databases to the XML-files.

It is necessary to include the following headers:

```
#include <epos_cposlmdatabasemanager.h>
#include <epos_cposlandmarkencoder.h>
#include <bautils.h> // - for BAFLUTILS
```

And link against:

```
LIBRARY eposlmdbmanlib.lib
LIBRARY eposlandmarks.lib
LIBRARY bafl.lib // - for BaflUtils
```

Source:

```
// we need RFs for some checks
RFs rFs;
User::LeaveIfError( rFs.Connect() );
CleanupClosePushL( rFs );

// backup folder
_LIT( KBackupFolder, "C:\\LmBackup\\" );

// check backup folder
TFileName backupFolder( KBackupFolder );
BaflUtils :: EnsurePathExistsL( rFs, backupFolder );

// for file name
_LIT( KReplace, "_" );
TBuf<1> replaceStr( KReplace );

// db manager
CPosLmDatabaseManager* dbManager = CPosLmDatabaseManager :: NewL();
CleanupStack :: PushL( dbManager );

// protocol for the local DBs
_LIT( KFileProto, "file" );

// array, that contains URI of the local DBs
CDesCArray* dbUriList = dbManager->ListDatabasesLC( KFileProto );

// MIME type for import & export
_LIT8( KPosMimeTypeLandmarkCollectionXml, "application/vnd.nokia.landmarkcollection+xml" );

// Create the encoder to be used for exporting landmarks
CPosLandmarkEncoder* encoder = CPosLandmarkEncoder::NewL(KPosMimeTypeLandmarkCollectionXml);
CleanupStack :: PushL( encoder );

// array of the export items
RArray<TPosLmItemId> itemsArray;
CleanupClosePushL( itemsArray );

for( TInt i = 0; i < dbUriList->Count(); i++ )
{
   // open current database
   TPtrC currentUri = (*dbUriList)[i];
   CPosLandmarkDatabase* db = CPosLandmarkDatabase :: OpenL( currentUri );
   CleanupStack :: PushL( db );

   // iterator allows to read the database landmarks
   CPosLmItemIterator* iter = db->LandmarkIteratorL();
   CleanupStack::PushL( iter );

   // if iterator is not empty
   if( iter->NumOfItemsL() > 0 )
   {
      // fetch IDs of all landmarks
      itemsArray.Reset();
      iter->GetItemIdsL( itemsArray, 0, iter->NumOfItemsL() );

      // filename from URI: file://e:my.ldb -> C:\LmBackup\e_my.ldb
      TFileName fileName( backupFolder );
      fileName.Append( currentUri.Right( currentUri.Length() - 7 ) );
      fileName.Replace( backupFolder.Length() + 1, 1, replaceStr ); // : -> _

      // if file already exists - delete file
      if( BaflUtils :: FileExists( rFs, fileName ) )
         BaflUtils :: DeleteFile( rFs, fileName );

      // point out the file to export landmark data to
      encoder->SetOutputFileL( fileName );

      // execute export in batch mode
      ExecuteAndDeleteLD( db->ExportLandmarksL( *encoder, itemsArray,
                          CPosLandmarkDatabase::EIncludeCategories ) );

      // finalize encoding to complete export ( in batch mode )
      ExecuteAndDeleteLD( encoder->FinalizeEncodingL() );
   }
   CleanupStack :: PopAndDestroy( 2 ); // iter db
}
CleanupStack :: PopAndDestroy( 5 ); // rFs itemsArray encoder iDbList dbManager
```


(C) Copyright Nokia 2013. All rights reserved.

Internal Links

- How to manage local landmark databases
- Landmarks/web client example using Carbide.c++ and UI designer
- How to use Landmarks API
- How to select and show a landmark
- How to compact local landmark databases
- How to import landmarks from file to database
- Execution of landmark operations
- How to obtain and save current location
- Retrieving location information
- How to manage landmark categories

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.