



The following code snippet demonstrates how to obtain list of the available positioning modules.

Headers:

```
#include <lbs.h>
```

Libraries:

```
LIBRARY      lbs.lib
LIBRARY      baf1.lib //for CDesCArrayFlat

// Position server
RPositionServer posServer;
User :: LeaveIfError( posServer.Connect() );
CleanupClosePushL( posServer );

// get default module UID
TPositionModuleId defModuleUid;
User :: LeaveIfError( posServer.GetDefaultModuleId( defModuleUid ) );

// get count of available modules
TUint numOfModules = 0;
User :: LeaveIfError( posServer.GetNumModules( numOfModules ) );
if( numOfModules )
{
    // array for storing
    CDesCArrayFlat* itemArray = new ( ELeave ) CDesCArrayFlat( numOfModules );
    CleanupStack :: PushL( itemArray );

    _LIT( KDefault, "*" ); // default module mark
    _LIT( KGeneral, "-" ); // general module mark
    const TInt KMaxModuleNameLen = 128;
    for( TUint i = 0; i < numOfModules; i++ )
    {
        // read current module info
        TPositionModuleInfo moduleinfo;
        posServer.GetModuleInfoByIndex ( i, moduleinfo );

        // read current module name
        TBuf< KMaxModuleNameLen > moduleName;
        moduleinfo.GetModuleName( moduleName );

        // insert marker "*" or "-"
        moduleName.Insert( 0, moduleinfo.ModuleId() == defModuleUid ?
                           KDefault : KGeneral );

        // append to array
        itemArray->AppendL( moduleName );
    }
    ShowListL( itemArray );
    CleanupStack :: PopAndDestroy(); // itemArray
}

CleanupStack :: PopAndDestroy(); // posServer
```

ShowListL function can be implemented as follows:

Available_positioning_modules

Headers:

```
#include <aknlists.h>
```

Libraries:

```
LIBRARY      avkon.lib
LIBRARY      eikcoctl.lib
LIBRARY      eikctl.lib
```

```
void YourClass :: ShowListL( CDesCArray* anArray )
{
    // create CEikTextListBox instance, list
    CEikTextListBox* list = new( ELeave ) CAknSinglePopupMenuStyleListBox;

    // push list's pointer to CleanupStack.
    CleanupStack::PushL( list );

    // create CAknPopupList instance, popupList
    CAknPopupList* popupList = CAknPopupList::NewL( list,
                                                    R_AVKON_SOFTKEYS_OK_EMPTY,
                                                    AknPopupLayouts::EMenuWindow );

    // push popupList's pointer to CleanupStack.
    CleanupStack::PushL( popupList );

    // initialize listbox.
    list->ConstructL( popupList, CEikListBox::ELeftDownInViewRect );
    list->CreateScrollBarFrameL( ETrue );
    list->ScrollBarFrame()->SetScrollBarVisibilityL( CEikScrollBarFrame::EOff,
                                                    CEikScrollBarFrame::EAuto );

    // set listitems.
    CTextListBoxModel* model = list->Model();
    model->SetItemTextArray( anArray );
    model->SetOwnershipType( ELbmDoesNotOwnItemArray );

    // show popup list and then show return value.
    popupList->ExecuteLD();

    // pop the popupList's pointer from CleanupStack
    CleanupStack::Pop();

    // pop and Destroy the list's pointer from CleanupStack
    CleanupStack::PopAndDestroy();
}
```