

Archived:Loading DLL by RLibrary

 Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template `{{ReviewForRemovalFromArchive|user=~~~~~|write your reason here}}`.

Overview

This code snippet shows how to load a polymorphic DLL dynamically.

A static DLL is automatically loaded in the RAM when a program that uses it is started. It is also automatically unloaded when it is not needed anymore.

A polymorphic DLL is loaded by calling `RLibrary::Load()` and unloaded using `RLibrary::Close()`. Several polymorphic DLLs can show the same interface to their clients. This kind of DLLs are generally used by a framework to provide plug-in features.

MMP file

The following capabilities and libraries are required:

```
CAPABILITY None
LIBRARY euser.lib
```

Polymorphic DLL

- UID2 must be 0x1000008d in DLL's MMP file.
- DLL must have equal or greater CAPABILITIES than the loading process. Once loaded, DLL runs at the capability level of the loading process. Therefore, a DLL must have all capabilities required by all its client executables, even if the code within the DLL itself does not require some of these capabilities.
- Define VIRTUAL INTERFACE class that the polymorphic DLLs will implement
- DLL should have one EXPORTED static function that returns an instance of the interface's class.

Your DLL header

```
class MMyDll
{
public:
    virtual TInt Data() = 0;
};

class CMyDll : public MMyDll
{
public:
    IMPORT_C static MMyDll* NewL();
    virtual TInt Data();
};
```

DLL source

```
EXPORT_C MMyDll* CMyDll::NewL()
{
    return new CMyDll();
}
```

```
TInt CMyDll::Data()  
{  
    return 1;  
}
```

Loading DLL dynamically

```
#include <e32std.h>  
  
RLibrary library;  
  
// Load dll  
User::LeaveIfError(library.Load(_L("CMyDll")));  
  
// Find exported function  
TLibraryFunction NewL=library.Lookup(1);  
MMyDll* mydll=(MMyDll*) NewL();  
  
// Close the library  
library.Close();  
  
TInt value =mydll->Data();  
delete mydll;
```

Postconditions

The DLL is loaded dynamically.