
Subject: Linking to a .net DLL ?

Posted by [nixnixnix](#) on Tue, 26 Jul 2011 20:43:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

Can't find anything on this when I search these fora.

I would like to be able to use some functionality which is proprietary and only available as .net

My colleague can get what I want to compile into a DLL but I am wondering how I then can use that DLL in UPP.

Cheers,

Nick

Subject: Re: Linking to a .net DLL ?

Posted by [dolik.rce](#) on Wed, 27 Jul 2011 05:26:27 GMT

[View Forum Message](#) <> [Reply to Message](#)

I'm not an expert on .net or windows programming, but I have a feeling that what you need is available in Koldos Functions4U package in bazaar.

Best regards,
Honza

Subject: Re: Linking to a .net DLL ?

Posted by [nixnixnix](#) on Thu, 28 Jul 2011 18:09:10 GMT

[View Forum Message](#) <> [Reply to Message](#)

Thanks Dolik and to you too, Koldo.

Exactly what I was looking for

Nick

Subject: Re: Linking to a .net DLL ?

Posted by [nixnixnix](#) on Thu, 28 Jul 2011 21:12:46 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hey Koldo,

Should this work for a .NET compiled DLL? I am trying the following

```

AgiRadar::AgiRadar()
{
// connecting to .net DLL to use the following functions

// Public Shared Function testFunc1() As Integer
//   Return 123
// End Function

// Public Shared Function testFunc2(ByVal x As Double) As Double
//   Return x * x
// End Function

DI turbineSpeedCalc;

String myDllFolder = "D:\\temp\\test\\Debug\\";

////////////////////////////////////

double (*testFunc2)(double);

if (!turbineSpeedCalc.Load(AppendFileName(myDllFolder, "turbineSpeedCalc.dll")))
    throw Exc(Format(t_("% dll not found"), "turbineSpeedCalc"));

testFunc2 = (double (*)(double))turbineSpeedCalc.GetFunction("testFunc2");

if (!testFunc2)
    throw Exc(Format(t_("Function %s was not found in dll"), "testFunc2"));

double argOut = testFunc2(15.0);

String s = Format("answer is %f",argOut);
PromptOK(s);

////////////////////////////////////

int (*testFunc1)();

if (!turbineSpeedCalc.Load(AppendFileName(myDllFolder, "turbineSpeedCalc.dll")))
    throw Exc(Format(t_("% dll not found"), "turbineSpeedCalc"));

testFunc1 = (int (*)( ))turbineSpeedCalc.GetFunction("testFunc1");

if (!testFunc1)
    throw Exc(Format(t_("Function %s was not found in dll"), "testFunc1"));

double argOut2 = testFunc1();

```

```
s = Format("answer is %f",argOut2);
PromptOK(s);

}
```

is there a way to list the functions available in a DLL? I figure if I can get these simple functions to work then I should be able to pass structs and arrays to real functions.

It successfully finds and loads the DLL but fails to find either function.

Cheers,

Nick

Subject: Re: Linking to a .net DLL ?
Posted by [Novo](#) on Fri, 29 Jul 2011 03:27:59 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi nixnixnix,
testFunc1 and testFunc2 most likely have C++ mangling.
I think you can get a list of exported function using either "VC\bin\dumpbin.exe" or dependency walker.

UPP has a nice feature called dli files.
You can find an example here: uppsrc\Oracle\Oci8.dli

I hope this will help.

Subject: Re: Linking to a .net DLL ?
Posted by [koldo](#) on Sun, 31 Jul 2011 21:37:20 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello Nick

In addition, I am not expert in .NET Dll but as DI uses LoadLibrary function, it could only load .NET Dll plain not managed functions outside a class.

Subject: Re: Linking to a .net DLL ?
Posted by [nixnixnix](#) on Sun, 31 Jul 2011 22:51:13 GMT

Thanks.

Dependency walker does not see any functions in the DLL so am thinking there is something that my colleague has neglected to do that is akin to publishing the functions. I have never used DLLs before so am not really sure what am talking about. He is on vacation just now so I will have to ask him when he gets back.

Cheers,

Nick

EDIT: so I just tried linking into the DLL with MS VC++ express and a piece of code from <http://support.microsoft.com/kb/953836>

With some small adjustment I can trigger the function `turbineSpeedCalc::testFunc1`. The main difference appears to be that `Functions4U` allows me to access only global functions within the DLL whereas I appear to need access to a class. If I can make an instance of the class that is within the DLL then I should be able to access its functions.

Subject: Re: Linking to a .net DLL ?

Posted by [JeyCi](#) on Mon, 17 Mar 2025 13:25:02 GMT

[View Forum Message](#) <> [Reply to Message](#)

Why compiling similar example, - getting error: 'DI' was not declared in this scope though I have done everything like here, but cannot declare DI Plugin; object in Upp `CONSOLE_APP_MAIN`.

p.s.

Details: using `upp-v.13664 x32`, in Console Project (with Core package in package organizer).

DLL was made from C#.NET-source:

```
using System;
using System.IO;
using System.Runtime.InteropServices;
using RGiesecke.DllExport;
```

```
namespace Plugins
```

```
{
    public class Plugins
    { // CallingConvention.StdCall
      [DllExport("OnPluginStart", CallingConvention = CallingConvention.Cdecl)]
      public static bool OnPluginStart()
      {
        using (var file = new StreamWriter(@"c:\pluginLog.txt", true))
        {
```

```

        file.WriteLine("OnPluginStart");
    }
    return true;
}
}
}

```

compiled with CMD Line:

Quote:C:\Windows\Microsoft.NET\Framework\v3.5\csc.exe /t:library /platform:x86
 /r:C:\Test\UnmanagedExports\lib\net\RGiesecke.DllExport.Meta data.dll /out:C:\Plugins.dll
 C:\Plugins.cs

MY .dli

FN(bool, OnPluginStart,(void))

Another problem, In Upp CONSOLE_APP_MAIN with

```

bool nn= PLUGINS().Load();
Cout() << nn << '\n';
- getting false.
And with
bool retval= PLUGINS().Load("C:\\Plugins.dll");
- Upp CONSOLE_APP_MAIN crashes

```

P.P.S.

Sorry, if am not right posting to this topic - just my problem seemed to be simply another problem under the same title...

Subject: Re: Linking to a .net DLL ?

Posted by [koldo](#) on Mon, 17 Mar 2025 15:21:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

Dear JeyCi

Maybe the confusion comes from the fact that dli files are a U++ feature, but the DI class is a UppHub/Functions4U feature.

You have some doc about DI here.

Subject: Re: Linking to a .net DLL ?

Posted by [JeyCi](#) on Mon, 17 Mar 2025 17:23:46 GMT

[View Forum Message](#) <> [Reply to Message](#)

Koldo, thanks for your appointment where can I find class DI, but I looked it through and I am not sure that I need to include it in my_app in general. Anyway, what do you think - if I have already

```
#define dllFILENAME "C:\\Plugins.dll"
#define DLIMODULE PLUGINS
#define DLIHEADER <my_use_DLL_cs/Plugins.dli>
#define dllCALL LNPUBLIC
```

then, as I understand, I don't need to put dll_path to T_PLUGINS::Load() once again, because it is already defined by #define-macros, I think. But why could this

```
bool n= PLUGINS().Load();
```

```
Cout() << n << '\n';
```

```
PLUGINS().Force();
```

give false to Console ?? - I cannot load my dll PLUGINS().OnPluginStart() at all in

```
if (PLUGINS().Load())
```

```
{
    bool MSG = PLUGINS().Load().OnPluginStart();
    LOG(MSG);
    Cout() << MSG << '\n';
}
```

with error

Quote:error: request for member 'OnPluginStart' in '(& PLUGINS())->T_PLUGINS::Load()', which is of non-class type 'bool'

Can it be due to twice-calling Load() ?? how to avoid it (if need) ? Anyway,just PLUGINS().Load(); seems to be not working - with PLUGINS().Load("C:\\Plugins.dll"); crashes here, without path given as argument - results false of Load()-method usage.

Subject: Re: Linking to a .net DLL ?

Posted by [JeyCi](#) on Mon, 17 Mar 2025 17:55:54 GMT

[View Forum Message](#) <> [Reply to Message](#)

Ok, I see in Core/dli.h

```
DLLTYPE& DLIMODULE()
```

```
{
    DLLTYPE& out = COMBINE(DLIMODULE, _());
    out.Load();
    return out;
}
```

therefore I just use

```
bool n= PLUGINS();
```

```
Cout() << n << '\n';
```

- result is false.

Koldo, SORRY :) Works

```
bool n= PLUGINS;
```

- result is true.

Still

```
CONSOLE_APP_MAIN
```

```
{
    StdLogSetup(LOG_COUT|LOG_FILE);
```

```

{
    bool n= PLUGINS;
    Cout() << n << '\n';
    //DLOG(AsString(n));
}
{
    bool (*testFunc2)();
    testFunc2 = (bool (*)())PLUGINS().OnPluginStart;
    Cout() << Value(*testFunc2) << '\n';
}
if (PLUGINS)
{
    bool MSG = PLUGINS().OnPluginStart();
    //LOG(MSG);
    Cout() << MSG << '\n';
}
}
- returns
Quote:true
false

```

:(without creating "c:\pluginLog.txt" in dll's function OnPluginStart exported ...

p.s. added archive with .net-recompiled dll (with CallingConvention.Cdecl to avoid doing anything with mangled name for the export function)

File Attachments

1) [Plugins.zip](#), downloaded 149 times

Subject: Re: Linking to a .net DLL ?

Posted by [JeyCi](#) on Thu, 05 Jun 2025 17:43:47 GMT

[View Forum Message](#) <> [Reply to Message](#)

msdn example of COM-Client and NET-Server, that is NET-dll in COM-container, created in VS as NET Class Library...

registered with regasm "c:\LoanLib.dll" /tlb:LoanLib.tlb /codebase f:_code\ClassLibraryForCom (better previously sign it with Strong Name Key) - still gives C1308-error: linking assemblies is not supported - A .netmodule [1][2] is allowed as input to the linker, but an assembly is not. ... here was a discussion - I'm curious how does it work now

P.S.

Well, if NET-comdll is built correctly and registered with regsvr32 (regasm no more needed) - it will work in C++ with CoCreateInstance as well
