

Forum:[Hack Forums](#)

Thread: [Dynative | Dynamically load and call native functions \(like Aeonhack did\)](#)

gigajew - November 18, 2017, 8:42 pm

Here's Dynative, much like Aeonhack's `CreateAPI<T>()` function it can map native function pointer to delegates, and get address of (or load) modules. In my implementation you can choose not to specify the string name of the function, but ONLY if the delegate name matches the function name.

Methods:

```
Code public static IntPtr GetModule(string name)
public static T GetFunction<T>(IntPtr module, string name)
public static T GetFunction<T>(IntPtr module)
```

Example usage:

Code static class Program

```
{
    // your native function delegate
    delegate DialogResult MessageBoxA(IntPtr hwnd, string text, string title, MessageBoxButtons
buttons);
    // user32.dll library
    static IntPtr user32 = Dynative.GetModule("user32.dll");
    // the MessageBoxA function, mapped to our delegate
    static MessageBoxA msgbox = Dynative.GetFunction<MessageBoxA>(user32);
    static void Main()
    {
        // invoke our MessageBoxA function
        msgbox.Invoke(IntPtr.Zero, "Hello world", "title", MessageBoxButtons.OK);
    }
}
```

[Download source code of: Dynative.cs](#)

EZLib - November 18, 2017, 10:03 pm

This is perfect, thank you so much for this. I am definitely integrating this into my project by tomorrow.

0xDEAD10CC - November 20, 2017, 11:38 am

You should check the return of `GetProcAddress()` with `IntPtr.Zero` before you attempt to use it, and probably throw an exception or return a bool for failure in that case. There's no point in using 'as' for the cast as pretty much everything directly or indirectly derives from `System.Object`... Just cast it if you're going to box/unbox.

Code (T)(object)...

```
Code public static Delegate GetDelegate<T>(IntPtr module, string name)
```

```
{
    var ptr = GetProcAddress(module, name);
    if (ptr == IntPtr.Zero) throw new ArgumentException("Function could not be found");
    return Marshal.GetDelegateForFunctionPointer(ptr, typeof(T));
}
```

With boxing and unboxing there's also `DynamicInvoke`.

And for your P/Invokes I would also probably try and keep those names synonymous with the actual native methods rather than renaming them in your source to something else for properness.

I would probably also just call LoadLibrary() directly as there's no real need to check for loaded modules. A call to LoadLibrary won't do any harm and will only just increase the reference count, therefore, if it's a module that may become unloaded by user code somewhere else in the program, if you called LoadLibrary directly for your own purpose, it also wouldn't break future calls that rely on the module being loaded in memory, so there's that too.

gigajew - November 20, 2017, 6:42 pm

Reason I check for loaded modules is to avoid loading them twice (if that's even a thing) -- but yeah

0xDEAD10CC - November 22, 2017, 4:24 pm

(Nov 20, 2017 07:42 PM)gigajew Wrote: [\[CITE\]](#) Reason I check for loaded modules is to avoid loading them twice (if that's even a thing) -- but yeah

It's not a thing, read the docs for LoadLibrary(). There's really very little reason not to just call it directly as many will suggest too throughout stackoverflow, but it makes sense. Although I still see many that do stuff like this.

Quote:The system maintains a per-process reference count on all loaded modules. Calling LoadLibrary increments the reference count. Calling the FreeLibrary or FreeLibraryAndExitThread function decrements the reference count. The system unloads a module when its reference count reaches zero or when the process terminates (regardless of the reference count).

Washi - November 23, 2017, 12:13 am

Regarding the exception handling, instead of throwing an ArgumentException that just tells you that it failed, you can also throw a new Win32Exception to get more information on why the GetProcAddress function failed, since this function updates the value of the GetLastError function :).

[https://msdn.microsoft.com/en-us/library/windows/desktop/ms683212\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms683212(v=vs.85).aspx)

gigajew - November 23, 2017, 6:17 am

throw new Win32Exception(Marshal.GetLastWin32Error());

And set SetLastError to true on the functions