

Forum:[Hack Forums](#)

Thread: [Create and Execute C# code at runtime, without creating external processes or files](#)

jujuger - March 11, 2017, 3:26 pm

Now this is extremely strange, but I want to create and execute code at runtime, but the thing is, I want to create and execute the code at runtime INSIDE SAID PROGRAM, without compiling again OR creating an external process or file.

Have the program create itself at runtime

Impossible?

background:

Spoiler ([Click to View](#)) I am trying to create the most realistic genetic-algorithm for evolution.

gigajew - March 11, 2017, 4:46 pm

All of my what.gif

Try to explain what you're trying to do again

BahNahNah - March 11, 2017, 5:18 pm

Yes, you can do this with CodeDom to compile the code in memory and reflection to run it.

.cYRil - March 11, 2017, 5:33 pm

As BahNahNah said you can do it with CodeDom make sure set GenerateExecutable (CompilerParameters) property to false.

Zettabit - March 11, 2017, 5:54 pm

(Mar 11, 2017 06:33 PM)HellGate Wrote: ([CITE](#)) As BahNahNah said you can do it with CodeDom make sure set GenerateExecutable (CompilerParameters) property to false.

Even setting GenerateExecutable to false and GenerateInMemory to true, it will still create a file on disk.

@OP, if you absolutely don't want any files to be created on the disk, you're going to have to look for alternatives. If it's not an issue, then this would be your best solution.

jujuger - March 11, 2017, 6:18 pm

(Mar 11, 2017 06:54 PM)Zettabit Wrote: ([CITE](#)) Even setting GenerateExecutable to false and GenerateInMemory to true, it will still create a file on disk.

@OP, if you absolutely don't want any files to be created on the disk, you're going to have to look for alternatives. If it's not an issue, then this would be your best solution.

The IO write is a problem because I need the program to iterate thousands of times per second. It needs to be speedy.

Is it possible?

ShadowCloudGH - March 11, 2017, 11:22 pm

This is interesting, you could potentially compile from a string as per the previous suggestion using

codeDom, however, I'm pretty sure you'll end up with files on disk in terms of temp files, something like this from stackoverflow :

```
Code using (Microsoft.CSharp.CSharpCodeProvider foo =
    new Microsoft.CSharp.CSharpCodeProvider())
{
    var res = foo.CompileAssemblyFromSource(
        new System.CodeDom.Compiler.CompilerParameters()
        {
            GenerateInMemory = true
        },
        "public class FooClass { public string Execute() { return \"output!\";}}\"
    );

    var type = res.CompiledAssembly.GetType("FooClass");

    var obj = Activator.CreateInstance(type);

    var output = type.GetMethod("Execute").Invoke(obj, new object[] { });
}
```

Do you really need the part that get compiled dynamically to be C#? Or could you look at using something like a python script and invoke python to run the script so you don't have a compilation issue?

w4zt3d - March 12, 2017, 1:30 am

Compile it with CodeDom like tghe others explained but Invoke the EntryPoint of the program thats nicer xD

ForceJew - March 12, 2017, 3:31 am

<https://hackforums.net/showthread.php?tid=5419995>

Check it out op

Marvincmarvin. - March 14, 2017, 12:37 pm

Use [Reflection.Emit](#) to write IL to an assembly in memory and invoke said assembly.

severos - March 15, 2017, 3:04 am

Would be awesome if you share your results when done OP

NyQuil - March 21, 2017, 4:53 pm

not sure if already replied but an example <http://pastebin.com/5QzCKckB>

Washi - March 23, 2017, 10:44 am

The [CompilerParameters.GenerateInMemory](#) property does not exactly do what its name implies. Even though you do not have to specify a path to the output file, the file will be written to a temporary directory instead, loaded as a [System.Reflection.Assembly](#) and put in the [CompilerResults.CompiledAssembly](#) property, hence creating the illusion of being it generated in memory. While this is often enough for the normal use of CodeDom's compiler services, for your case it might slow down a lot since a disk performs its read and write operations a lot slower than memory does.

Since you want to create a "realistic" genetic algorithm, what you might wanna look for is some kind of incremental compilation, where you keep changing your source code according to the feedback you get and recompile on the fly. Have a look at [Roslyn](#), the open source compiler platform

developed and released by Microsoft. Not only does it allow you to truly compile to memory, but also allows you to parse and change C# syntax to be compiled using syntax trees, which might be really useful in your case.

Here is a tutorial that I found on the web that might help you out a bit:

<http://www.tugberkugurlu.com/archive/compiling-c-sharp-code-into-memory-and-executing-it-with-roslyn>

Good luck!