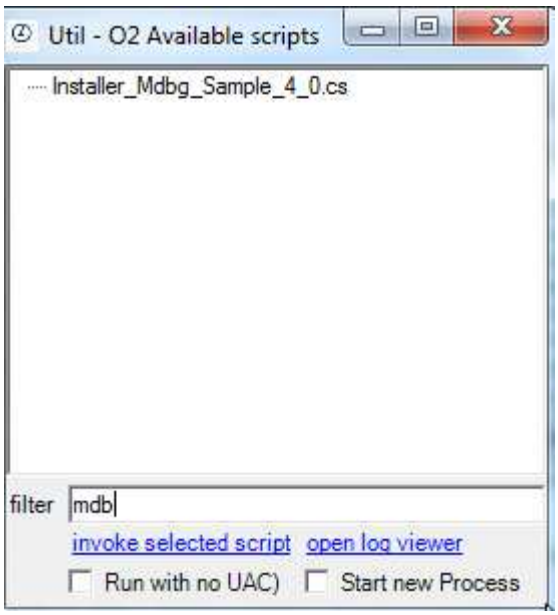
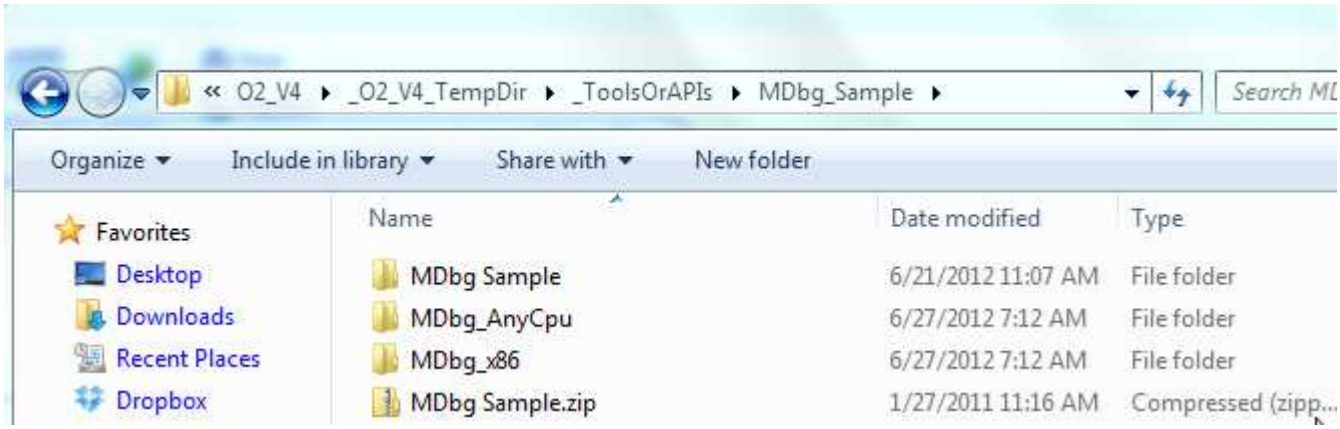
Scripting MDbg

This is using the MDbg whose source code is made available

Install it using Installer_Mdbg_Sample_4_0.cs

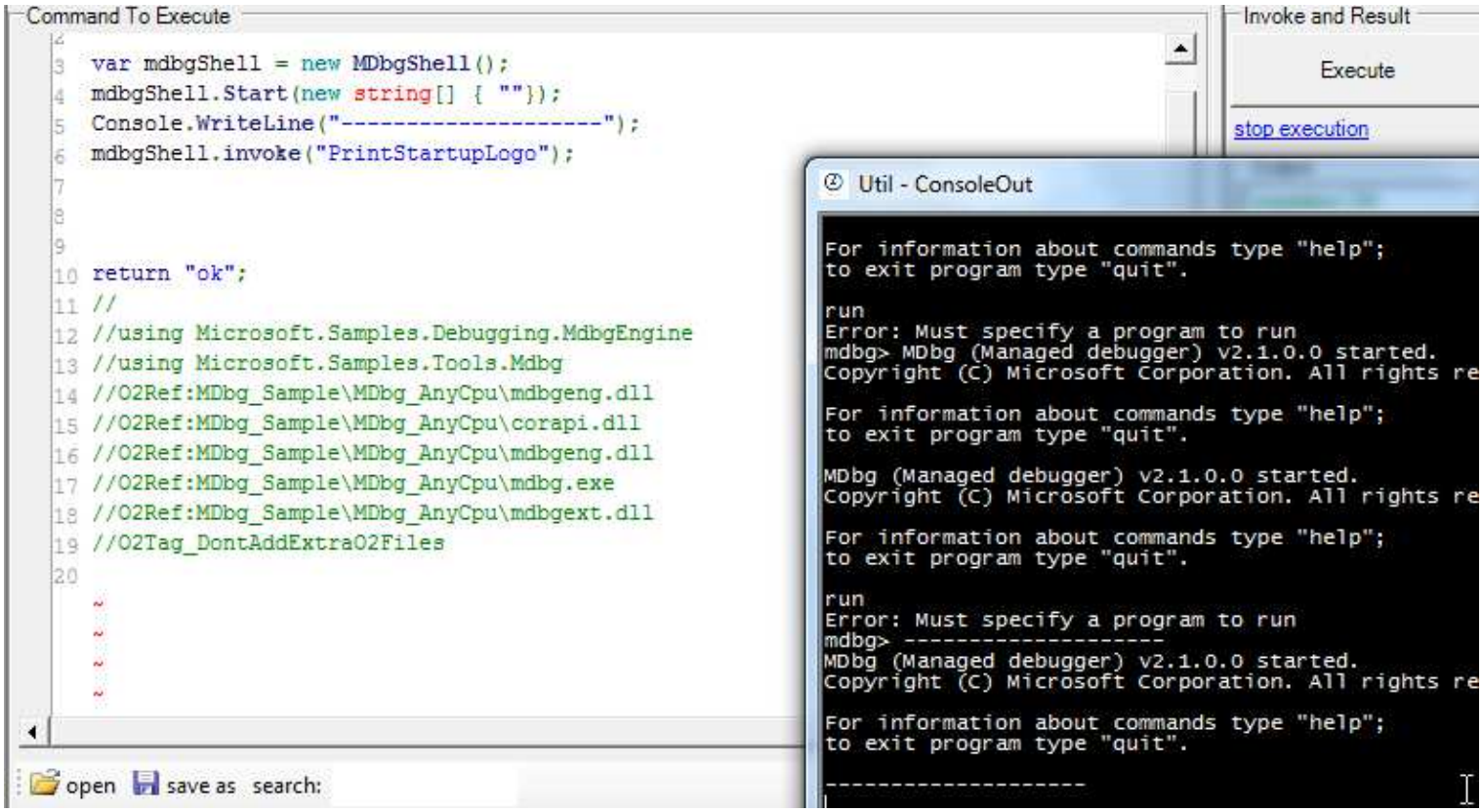


which will download, compile and install mdbg into



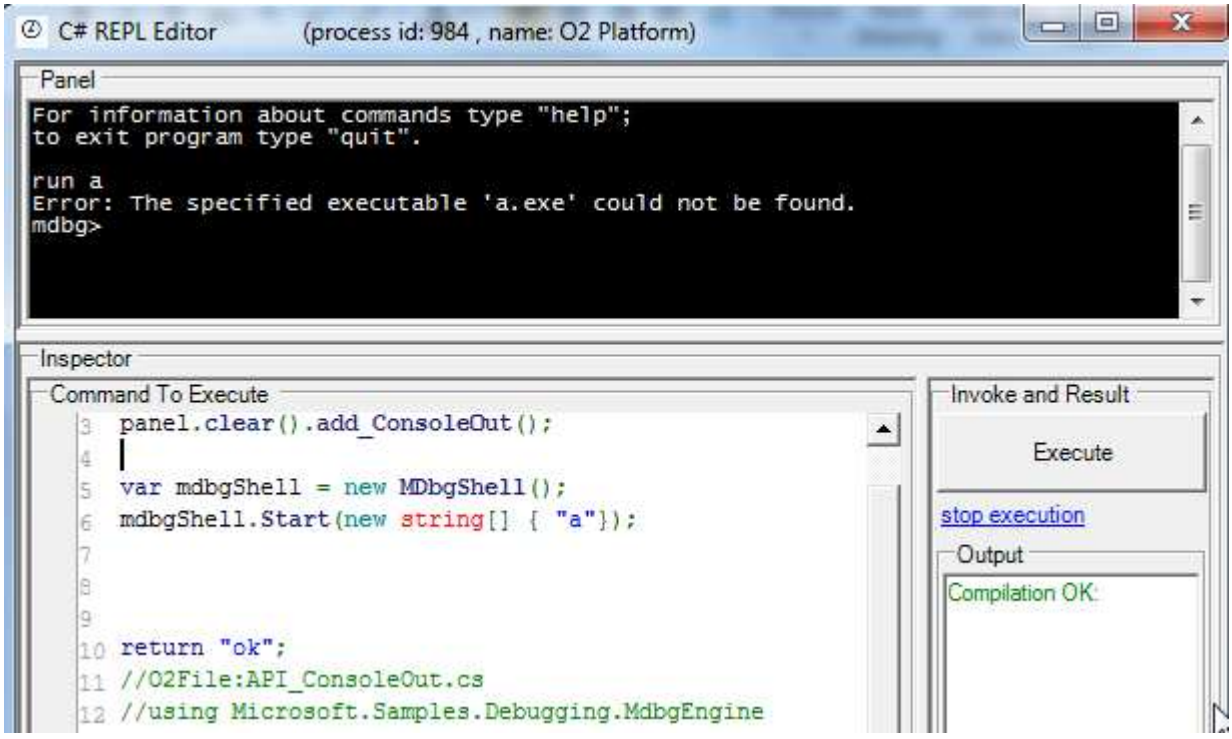
Script to start it (showing result in ConsoleOut)

```
var mdbgShell = new MDbgShell();
mdbgShell.Start(new string[] { ""});
Console.WriteLine("--------------------");
mdbgShell.invoke("PrintStartupLogo");


return "ok";
//
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg_AnyCpu\corapi.dll
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbg.exe
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbgext.dll
//O2Tag_DontAddExtraO2Files
```



You can also add the console out the O2 REPL's panel

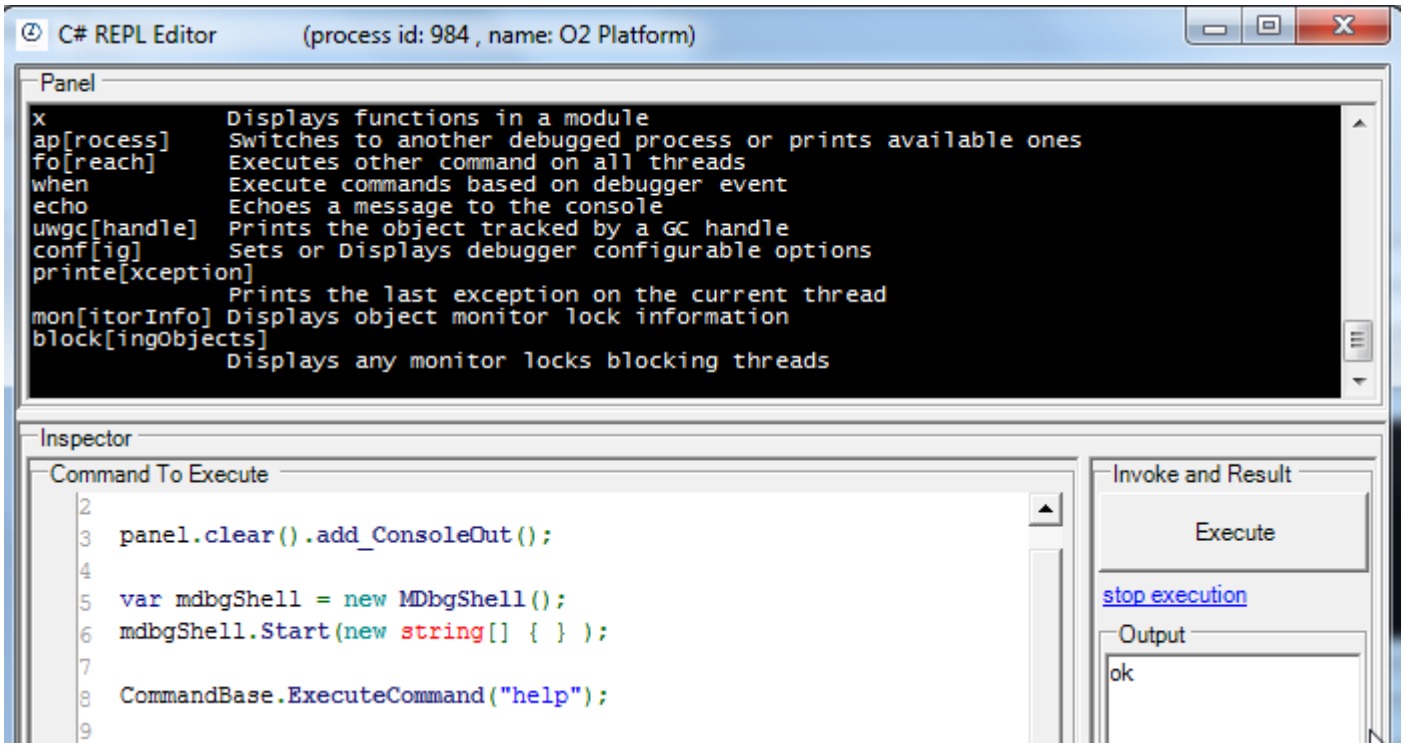
```
panel.clear().add_ConsoleOut();

var mdbgShell = new MDbgShell();
mdbgShell.Start(new string[] { "a"});

return "ok";
//O2File:API_ConsoleOut.cs
```



Help Command

```
var mdbgShell = new MDbgShell();
```

```
mdbgShell.Start(new string[] { } );

CommandBase.ExecuteCommand("help");
```



You can also set-up the objectly directly

```
var mdbgCommands = "mdbg.exe".assembly().type("MdbgCommands");

panel.clear().add_ConsoleOut();

var mdbgShell = new MDbgShell();
//mdbgShell.Start(new string[] { } );
var initialCommands = new string[] {"/?"};
mdbgShell.IO = new MDbgIO(mdbgShell, initialCommands);
CommandBase.Shell = mdbgShell;
mdbgShell.Debugger = new MDbgEngine();
mdbgCommands.invokeStatic("Initialize");

CommandBase.ExecuteCommand("help");
CommandBase.ExecuteCommand("a");


//var whenHandler =  mdbgCommands.method("WhenHandler"); // use if need to set-up OnCommandExecuted
//mdbgShell.OnCommandExecuted += (sender,e)=> { "OnCommandExecuted".info();};
//mdbgShell.OnCommandExecuted += new CommandExecutedEventHandler(MdbgCommands.WhenHandler);

return "ok";

//O2File:API_ConsoleOut.cs
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg_AnyCpu\corapi.dll
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbg.exe
//O2Ref:MDbg_Sample\MDbg_AnyCpu\mdbgext.dll
//O2Tag_DontAddExtraO2Files.
```

Starting a process and seeing list of avaiable processes to attach

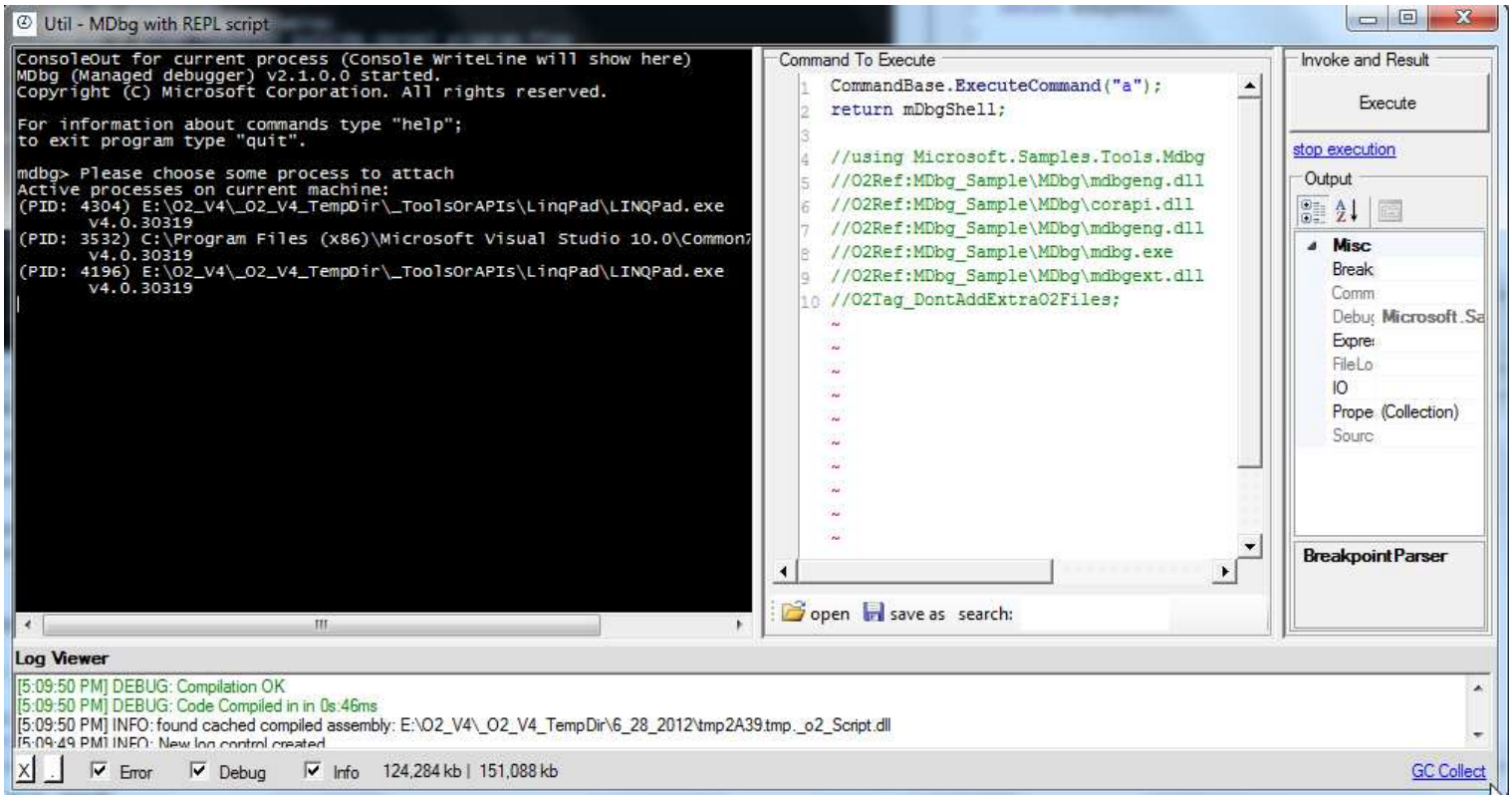
```
var mdbgCommands = "mdbg.exe".assembly().type("MdbgCommands");

var file = @"E:\O2_V4\_O2_V4_TempDir\_ToolsOrAPIs\LinqPad\LINQPad.exe";
var process = file.startProcess();
```

```
panel.clear().add_ConsoleOut();

var mdbgShell = new MDbgShell();
mdbgShell.Start(new string[] { } );

//See available processes to attach
CommandBase.ExecuteCommand("a");
process.closeInNSeconds(10);

return process.Id;

//O2File:API_ConsoleOut.cs
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Tag_DontAddExtraO2Files
```

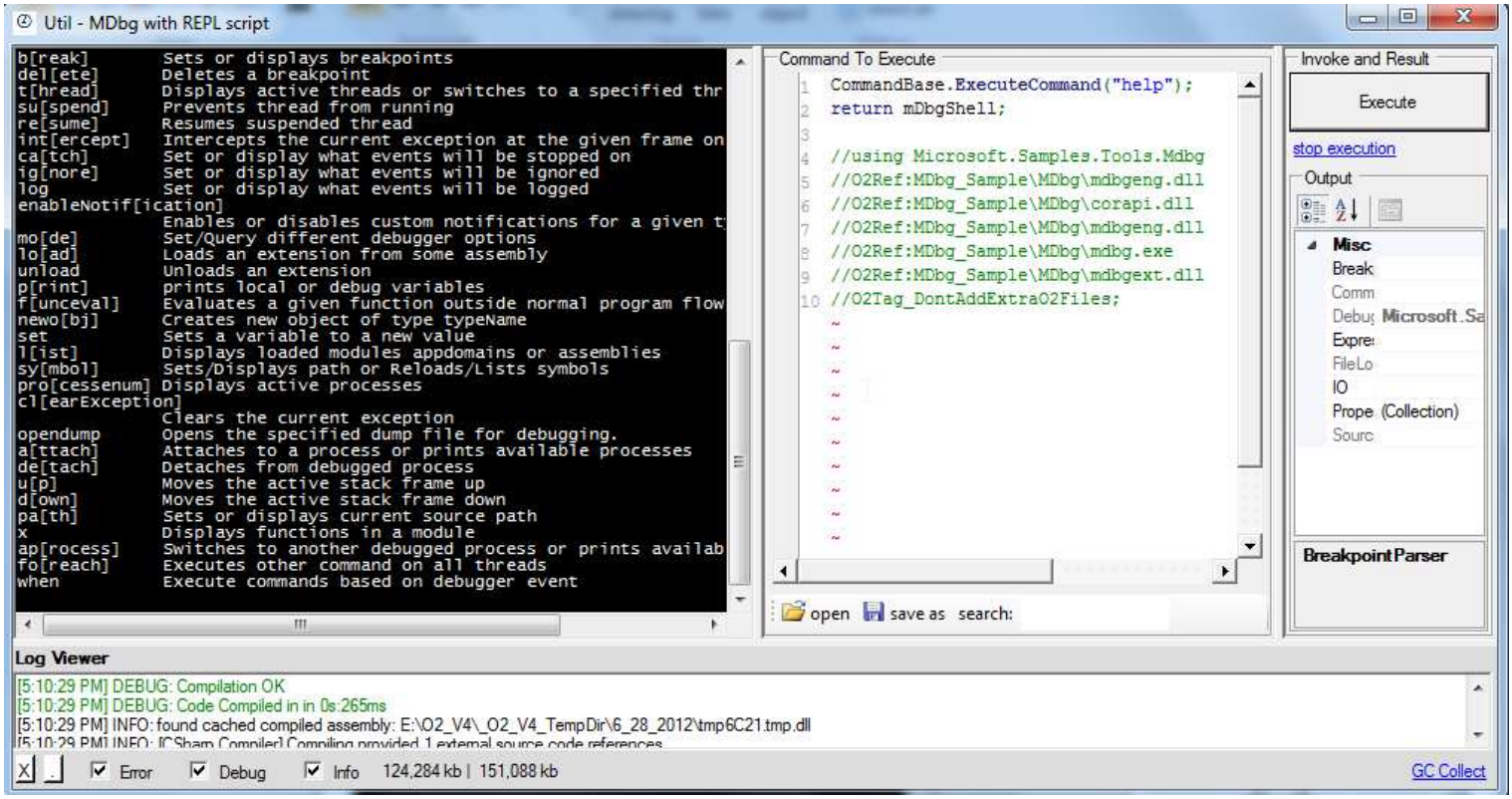
Nice util to dynamically execute MDbg commands:

```
//var topPanel = panel.clear().add_Panel();
var topPanel = "Util - MDbg with REPL script".popupWindow(1000,500).insert_LogViewer();
var consoleOut =  topPanel.clear().add_ConsoleOut();

var mdbgCommands = "mdbg.exe".assembly().type("MdbgCommands");
var mdbgShell = new MDbgShell();
try
{
      mdbgShell.Start(new string[] { } );
}
catch {}

topPanel.insert_Right().add_Script_Object(mdbgShell).Code =
@"CommandBase.ExecuteCommand(""a"");
return mDbgShell;

//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Tag_DontAddExtraO2Files;";
//See available processes to attach


//O2File:Scripts_ExtensionMethods.cs
//O2File:API_ConsoleOut.cs
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Tag_DontAddExtraO2Files
```

Which looks like this :



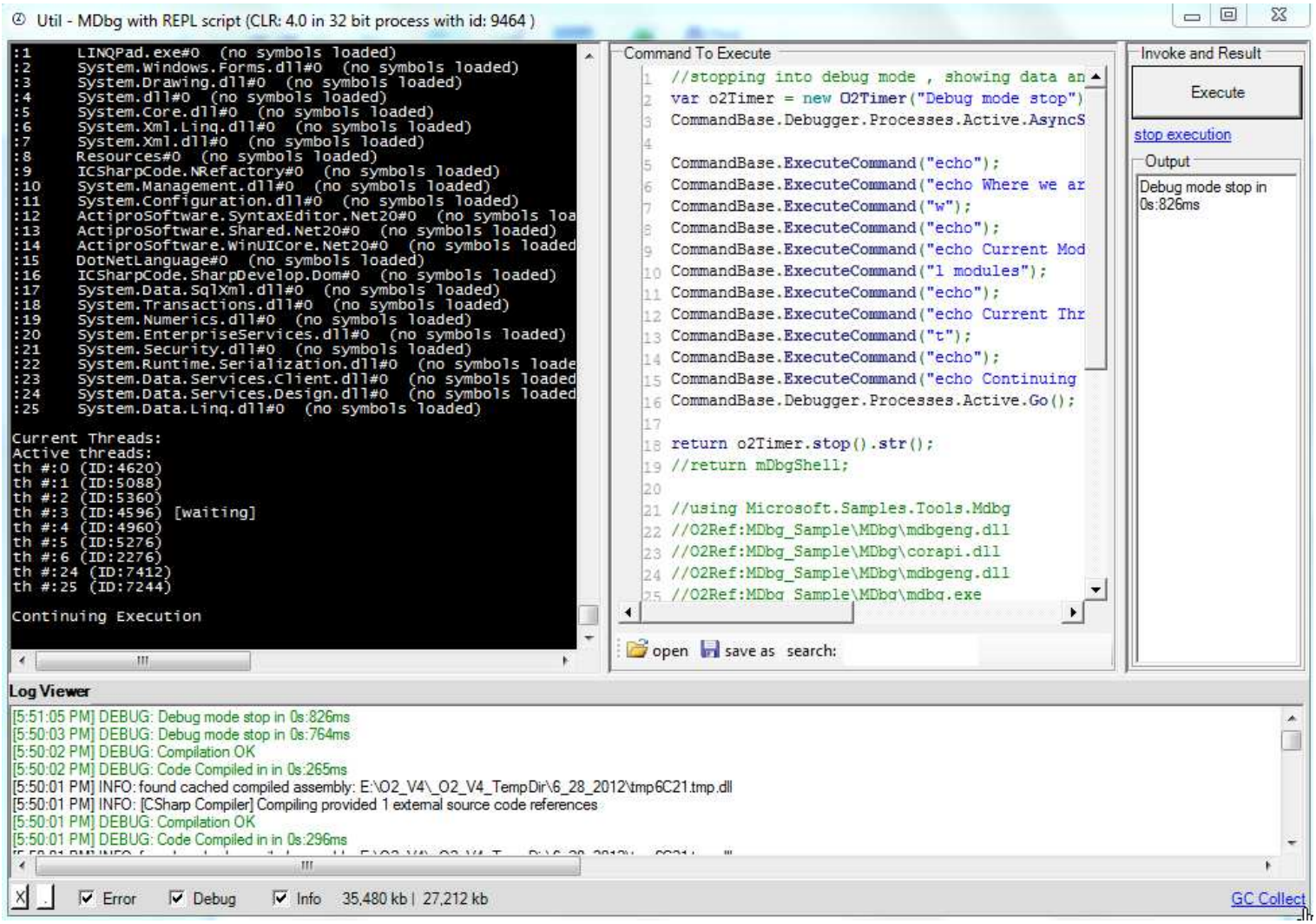
And like this after sending the "help" command to the in-memory instance of MDbd



Stopping into debug mode , showing data and continuing

```
//stopping into debug mode , showing data and continuing
var o2Timer = new O2Timer("Debug mode stop").start();
CommandBase.Debugger.Processes.Active.AsyncStop().WaitOne();

CommandBase.ExecuteCommand("echo");
CommandBase.ExecuteCommand("echo Where we are:");
CommandBase.ExecuteCommand("w");
```

```
CommandBase.ExecuteCommand("echo");
CommandBase.ExecuteCommand("echo Current Modules:");
CommandBase.ExecuteCommand("l modules");
CommandBase.ExecuteCommand("echo");
CommandBase.ExecuteCommand("echo Current Threads:");
CommandBase.ExecuteCommand("t");
CommandBase.ExecuteCommand("echo");
CommandBase.ExecuteCommand("echo Continuing Execution");
CommandBase.Debugger.Processes.Active.Go();

return o2Timer.stop().str();
//return mDbgShell;

//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
```



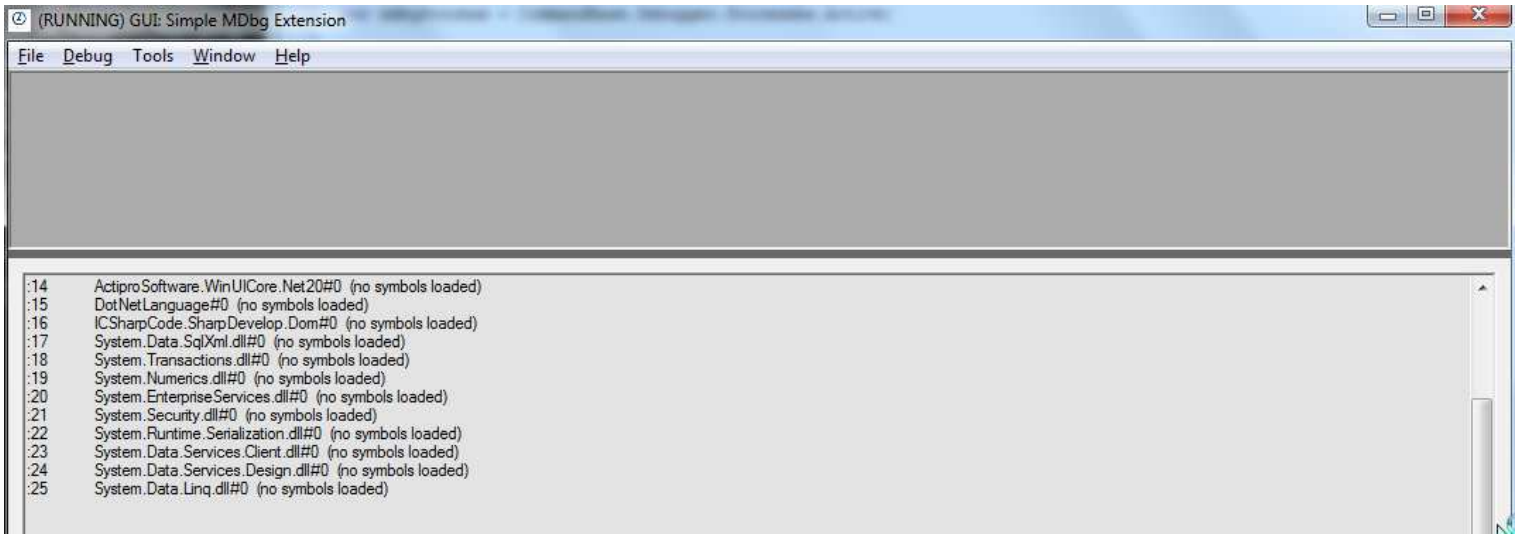
There is a GUI PoC included as an MDbd extension which can be loaded/opened like this (first time)

```
CommandBase.ExecuteCommand(@"load " + "gui.dll".assembly().Location);
//O2Ref:MDbg_Sample\MDbg\gui.dll
```

or like this (after it is loaded)

```
CommandBase.ExecuteCommand("gui");
```

The gui looks like this



.

.

Stuck with NotImplemented error. This code (executed after binded)

```
var o2Timer = new O2Timer("Debug mode stop").start();
var mdbgProcess = CommandBase.Debugger.Processes.Active;
O2Thread.staThread(
()=>{

                    CommandBase.Debugger.Processes.Active.AsyncStop().WaitOne();


                    MDbgThread t = mdbgProcess.Threads.Active;
                    MDbgFrame f=t.CurrentFrame;
                    foreach (MDbgValue v in f.Function.GetActiveLocalVars(f))
                                                {
                                                   Console.WriteLine(v.Name);
                                                   Console.WriteLine(v.Value);
                                                }
});
//return "ok";
//corProcess.details();
O2Thread.staThread(
       ()=>{
                    var objects = mdbgProcess.CorProcess.Objects;
                    //var corProcess = (ICorDebugProcess3)mdbgProcess.CorProcess.Raw;
                    //ICorDebugObjectEnum eobj = null;
                    //corProcess.EnumerateObjects(out eobj);
                    //var objects = mdbgProcess.CorProcess;
                    //objects.showInfo();
             });

return "ok";

//CommandBase.Debugger.Processes.Active.Go();

return o2Timer.stop().str();
//return mDbgShell;
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Debugging.CorDebug.NativeApi
//using Microsoft.Samples.Tools.Mdbg
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\raw.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Ref:MDbg_Sample\MDbg\NativeDebugWrappers.dll
```

Throws these errors:

[12:09:01 AM] ERROR: in staThread Unable to cast COM object of type 'System.__ComObject' to interface type 'Microsoft.Samples.Debugging.CorDebug.NativeApi.ICorDebugProcess'. This operation failed because the QueryInterface call on the COM component for the interface with IID '{3D6F5F64-7538-11D3-8D5B-00104B35E7EF}' failed due to the following error: No such interface supported

(Exception from HRESULT: 0x80004002 (E_NOINTERFACE)).
[12:09:01 AM] ERROR: in staThread Unable to cast COM object of type 'System.__ComObject' to interface type 'Microsoft.Samples.Debugging.CorDebug.NativeApi.ICorDebugProcess'. This operation failed because the QueryInterface call on the COM component for the interface with IID '{3D6F5F64-7538-11D3-8D5B-00104B35E7EF}' failed due to the following error: No such interface supported (Exception from HRESULT: 0x80004002 (E_NOINTERFACE)).

Also doesn't work:

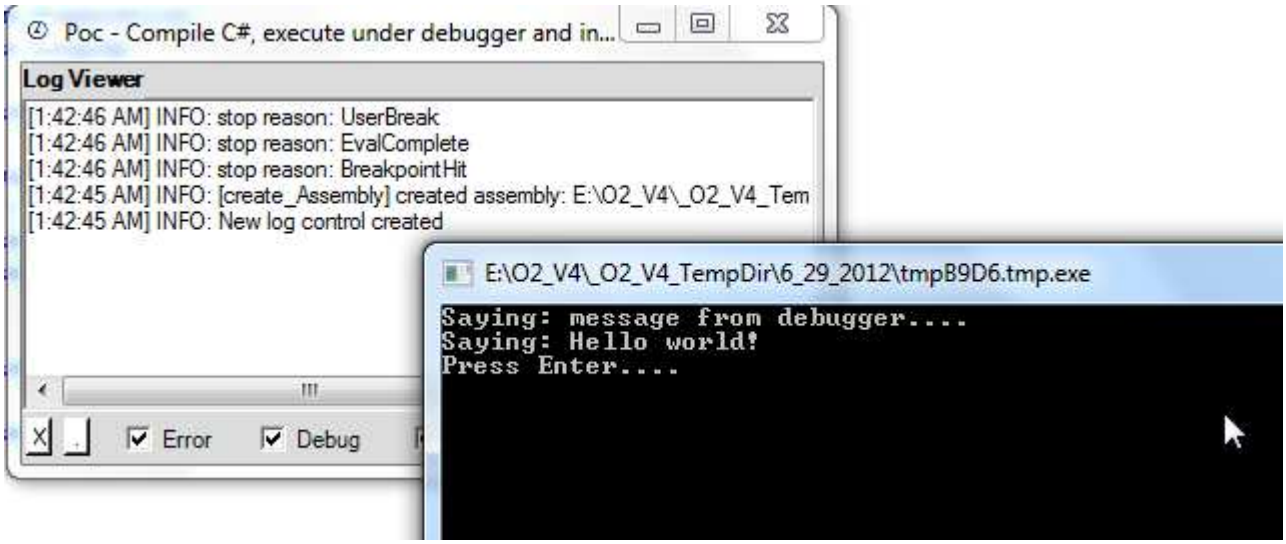
```
ICorDebugObjectEnum ppObjects = null;
debugger.activeProcess().CorProcess.Raw.EnumerateObjects(out ppObjects);
```

InnerException: at Microsoft.Samples.Debugging.CorDebug.NativeApi.ICorDebugProcess.EnumerateObjects(ICorDebugObjectEnum& ppObjects)

Poc - Compile C#, execute under debugger and invoke method from debugger

```
"Poc - Compile C#, execute under debugger and invoke method from debugger".popupWindow
(400,200).add_LogViewer();
var code = @"using System;
using System.Diagnostics;
public class Program
{
	public static void Main(String[] args)
	{
		Debugger.Break();
		Say(""Hello world!"");
		Console.WriteLine(""Press Enter...."");
		Console.ReadLine();
	}

	public static void Say(String str)
	{
		Console.WriteLine(""Saying: "" + str);
	}
}";

var exeToDebug =  code.createExe();

MDbgEngine debugger = new MDbgEngine();

//debugger.Options.StopOnLogMessage = true;

debugger.CreateProcess(exeToDebug, "", DebugModeFlag.Debug, null);

  while (debugger.Processes.Count > 0 && debugger.Processes.Active.IsAlive)
  {
    debugger.Processes.Active.Go().WaitOne();
    if (debugger.Processes.Count ==0)
       break;

       "stop reason: {0}".info(debugger.Processes.Active.StopReason);
    if (debugger.Processes.Active.StopReason is BreakpointHitStopReason)
    {
       var methodToInvoke = "Program.Say";
       var strToPassToDebuggee = "message from debugger....";

      // get function
      MDbgFunction func = debugger.Processes.Active.ResolveFunctionNameFromScope(methodToInvoke,
debugger.Processes.Active.AppDomains[0].CorAppDomain);

      CorEval eval = debugger.Processes.Active.Threads.Active.CorThread.CreateEval();
      eval.NewString(strToPassToDebuggee);

      debugger.Processes.Active.Go().WaitOne();
      CorValue corStrToPassToDebuggee = (debugger.Processes.Active.StopReason as
EvalCompleteStopReason).Eval.Result;

      eval.CallFunction(func.CorFunction, new CorValue[] { corStrToPassToDebuggee });

    }
}
"Debug session ended".debug();
return "all done";


//using Microsoft.Samples.Debugging.CorDebug
//using Microsoft.Samples.Debugging.MdbgEngine
```

```
//using Microsoft.Samples.Tools.Mdbg

//O2File:API_ConsoleOut.cs
//O2File:_Extra_methods_Roslyn_API.cs

//O2Ref:Roslyn.Compilers.dll
//O2Ref:Roslyn.Compilers.CSharp.dll

//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Tag_DontAddExtraO2Files
```



.

.

.

.

Creating MDbg ExtensionMethods

Get Action process and stop after some seconds

```
    public static class MDbg_ExtensionMethods
    {
            public static MDbgProcess activeProcess(this MDbgEngine engine)
            {
                  return engine.Processes.Active;
            }

            public static MDbgEngine process_StopInNSeconds(this MDbgEngine engine, int seconds)
            {
                  O2Thread.mtaThread(
                        ()=>{
                                    engine.sleep(seconds * 1000);
                                    engine.activeProcess().Kill();
                              });
                  return engine;
            }
     }
```

get Modules and modules names (I.e the full path to the assemblies loaded)

```
            public static List<MDbgModule> modules(this MDbgEngine engine)
            {
```

```
                var modules = new List<MDbgModule>();
                foreach(MDbgModule module in engine.activeProcess().Modules)
                        modules.Add(module);
                return modules;
        }

        public static List<string> names(this List<MDbgModule> modules)
        {
                return (from module in modules
                                select module.CorModule.Name).toList();
        }
```

get specific module

```
var msCorLib = debugger.module("mscorlib");
return msCorLib;
```

```
public static MDbgModule module(this  MDbgEngine engine, string moduleName)
{
        return engine.activeProcess().Modules.Lookup(moduleName);
}
```

get types and methods

```
public static class MDbg_ExtensionMethods_Types
{
        public static List<MetadataType> types(this MDbgModule module)
        {
                var types = new List<MetadataType>();
                foreach( MetadataType type in module.Importer.DefinedTypes)
                        types.add(type);
                return types;
        }

        public static MetadataType type(this MDbgModule module, string typeName)
        {
                return (from type in module.types()
                                where type.Name == typeName
                                select type).first();
        }
}

public static class MDbg_ExtensionMethods_Methods
{
        public static List<MethodInfo> methods(this MetadataType type)
        {
                //the bindingAttr is not used
                return type.GetMethods(BindingFlags.Default).toList();
        }

        public static List<MethodInfo> methods(this MetadataType type, string methodName)
        {
                return (from method in type.methods()
                                where method.Name == methodName
                                select method).toList();
        }

        public static MethodInfo method(this MetadataType type, string methodName)
        {
                return (from method in type.methods()
                                where method.Name == methodName
                                select method).first();
        }
}
```

Invoking ManagedMethods in Current Thread (in this case System.Console.WriteLine)

```
//"Poc - Compile C#, execute under debugger and invoke method from debugger".popupWindow
(400,200).add_LogViewer();
"exeToDebug".o2Cache(null);
var exeToDebug = "exeToDebug".o2Cache<string>(()=>{
                                                                "Creating temp
```

```
exe".info();

System;

System.Diagnostics;

Program

static void Main(String[] args)


       Debugger.Break();

(""Hello world!"");

       Console.WriteLine(""Press Enter...."");

       Console.ReadLine();


static void Say(String str)

       Console.WriteLine(""Saying: "" + str);


code.createExe();
```

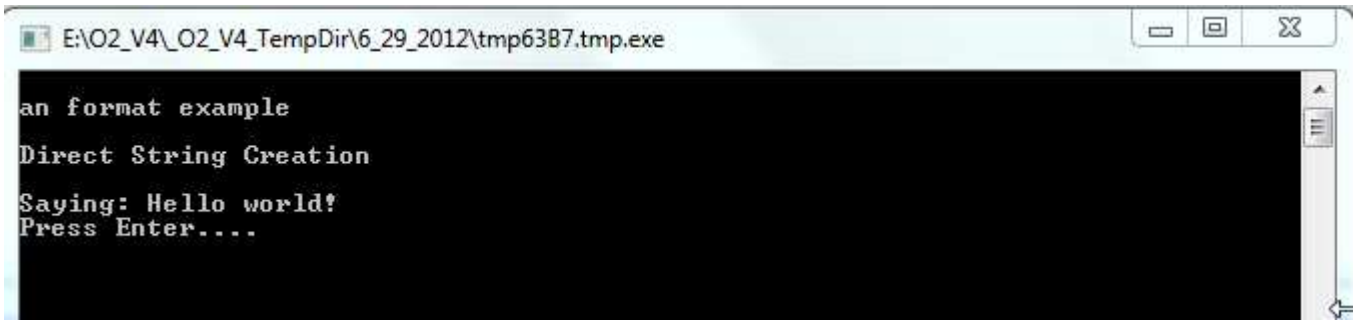
```
var code = @"using

using

public class

{
       public

       {

//

                Say

       }

       public

       {

       }
}";

return

});

MDbgEngine debugger = new MDbgEngine();

var process = debugger.CreateProcess(exeToDebug, "", DebugModeFlag.Debug, null);


  while (debugger.Processes.Count > 0 && debugger.Processes.Active.IsAlive)
  {
    debugger.Processes.Active.Go().WaitOne();
    if (debugger.Processes.Count ==0)
       break;

       "stop reason: {0}".info(debugger.Processes.Active.StopReason);

    if (debugger.Processes.Active.StopReason is BreakpointHitStopReason)
    {

              debugger.process_StopInNSeconds(27);

              var iCorDebugModule = debugger.modules().first().CorModule.Raw;

              //invoke System.Console.Write(format, object)
              var message = "this is a : {0}";
              var value = "formated";
              var msCorLib = debugger.module("mscorlib");
              var methods = msCorLib.type("System.Console").methods("Write").first();

              var mdbgFunction = process.ResolveFunctionName(msCorLib, "System.Console", "Write");

              CorEval eval = debugger.Processes.Active.Threads.Active.CorThread.CreateEval();
       eval.NewString(message);
       debugger.Processes.Active.Go().WaitOne();
       CorValue corStrToPassToDebuggee = (debugger.Processes.Active.StopReason as
EvalCompleteStopReason).Eval.Result;
       eval.NewString(value);
       debugger.Processes.Active.Go().WaitOne();
              CorValue corStrToPassToDebuggee2 = (debugger.Processes.Active.StopReason as
EvalCompleteStopReason).Eval.Result;
       eval.CallFunction(mdbgFunction.CorFunction, new CorValue[] { corStrToPassToDebuggee,
corStrToPassToDebuggee2 });
       debugger.Processes.Active.Go().WaitOne();

       //invoke System.Console.WriteLine()
          var writeLine = process.ResolveFunctionName(msCorLib, "System.Console", "WriteLine");
          var eval2 = debugger.Processes.Active.Threads.Active.CorThread.CreateEval();
          eval2.CallFunction(writeLine.CorFunction, new CorValue[] { });
              debugger.Processes.Active.Go().WaitOne();
    }
```

```
}

"Debug session ended".debug();
return "all done";


//using Microsoft.Samples.Debugging.CorDebug
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Tools.Mdbg

//O2File:API_ConsoleOut.cs
//O2File:_Extra_methods_Roslyn_API.cs

//O2Ref:Roslyn.Compilers.dll
//O2Ref:Roslyn.Compilers.CSharp.dll

//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Ref:MDbg_Sample\MDbg\raw.dll
//O2Tag_DontAddExtraO2Files

//O2File:MDbg_ExtensionMethods.cs
```

we can refactor the invoke workflow to make it easier:

```
//"Poc - Compile C#, execute under debugger and invoke method from debugger".popupWindow
(400,200).add_LogViewer();
"exeToDebug".o2Cache(null);
var exeToDebug = "exeToDebug".o2Cache<string>(()=>{
                                                                                    "Creating temp
exe".info();
                                                                                    var code = @"using
System;
                                                                                    using
System.Diagnostics;
                                                                                    public class
Program
                                                                                    {
                                                                                           public
static void Main(String[] args)
                                                                                           {

                                                                                    //
       Debugger.Break();
                                                                                                  Say
(""Hello world!"");
       Console.WriteLine(""Press Enter...."");
       Console.ReadLine();
                                                                                           }

                                                                                           public
static void Say(String str)
                                                                                           {
       Console.WriteLine(""Saying: "" + str);
                                                                                           }
                                                                                    }";
                                                                                    return
code.createExe();
                                                                            });

MDbgEngine debugger = new MDbgEngine();
var process = debugger.CreateProcess(exeToDebug, "", DebugModeFlag.Debug, null);


  while (debugger.Processes.Count > 0 && debugger.Processes.Active.IsAlive)
  {
    debugger.Processes.Active.Go().WaitOne();
    if (debugger.Processes.Count ==0)
```

```
        break;

        "stop reason: {0}".info(debugger.Processes.Active.StopReason);

    if (debugger.Processes.Active.StopReason is BreakpointHitStopReason)
    {

                debugger.process_StopInNSeconds(7);

                debugger.invokeMethod("mscorlib", "System.Console","WriteLine" );

                var param1 = debugger.create_String("an {0} example");
                var param2 = debugger.create_String("format");
                debugger.invokeMethod("mscorlib", "System.Console","Write", new CorValue[] {
param1,param2});

                debugger.console_WriteLine();
                debugger.console_WriteLine();

                debugger.invokeMethod("mscorlib", "System.Console","Write", "Direct {0}","String
Creation");

                debugger.console_WriteLine();
                debugger.console_WriteLine();
    }
}

"Debug session ended".debug();
return "all done";


//using Microsoft.Samples.Debugging.CorDebug
//using Microsoft.Samples.Debugging.MdbgEngine
//using Microsoft.Samples.Tools.Mdbg

//O2File:API_ConsoleOut.cs
//O2File:_Extra_methods_Roslyn_API.cs

//O2Ref:Roslyn.Compilers.dll
//O2Ref:Roslyn.Compilers.CSharp.dll

//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\corapi.dll
//O2Ref:MDbg_Sample\MDbg\mdbgeng.dll
//O2Ref:MDbg_Sample\MDbg\mdbg.exe
//O2Ref:MDbg_Sample\MDbg\mdbgext.dll
//O2Ref:MDbg_Sample\MDbg\raw.dll
//O2Tag_DontAddExtraO2Files

//O2File:MDbg_ExtensionMethods.cs
```



Invoking a method via its MetadataToken

```
var token = debugger.module("mscorlib").type("System.Console").method("Write").MetadataToken;
var function = debugger.module("mscorlib").CorModule.GetFunctionFromToken(token);
       debugger.invoke_Method(function, "testing: {0}", "123");
```

invoking a method from a MethodInfo

```
var methodInfo = debugger.module("mscorlib").type("System.Console").method("Write");
```

```
var module = debugger.process().ResolveClass(methodInfo.DeclaringType.FullName).Module;
var corFunction = module.GetFunctionFromToken(methodInfo.MetadataToken);
debugger.invoke_Method(corFunction,"testing: {0}", "123AAAAAAAAA");
```

or after refactoring

```
var methodInfo = debugger.module("mscorlib").type("System.Console").method("Write");
methodInfo.invoke_Method(debugger, "1234 {0}","45");
```

since we are using a MethodInfo we can also get its value using normal reflection:

```
var methodInfo = typeof(Console).method("Write");
methodInfo.invoke_Method(debugger,"testing: {0}", "123AAAAAAAAA");
```

Multiple invokes:

```
typeof(Console).method("WriteLine")
               .invoke_Method(debugger);
typeof(Console).method("Write")
               .invoke_Method(debugger,"from: {0}", "reflection , press enter");
typeof(Console).method("ReadLine")
               .invoke_Method(debugger);
```

Invoking a method with a specific signature

```
typeof(Console).method_bySignature("Void WriteLine(System.String)")
               .invoke_Method(debugger,"Another from Reflection");
```

Invoking with a specific parameter

```
            var corEval = debugger.corEval();
            char charValue = 'B';
            var charType = debugger.process().ResolveClass(charValue.typeFullName());
            CorValue val = corEval.CreateValue(CorElementType.ELEMENT_TYPE_CHAR , charType);
            val.CastToGenericValue().SetValue(charValue);

            typeof(Console).method_bySignature("Void WriteLine(Char)")
                           .invoke_Method(debugger, new CorValue[] { val});
```

Trying to get the value of a string

```
            var newString = debugger.create_String("a new string");
            try { "CastToArrayValue          : {0}".info(newString.CastToArrayValue());
             }           catch(Exception ex)       {                   ex.log();          }
            try { "CastToBoxValue            : {0}".info(newString.CastToBoxValue());
             }           catch(Exception ex)       {                   ex.log();          }
            try { "CastToGenericValue  : {0}".info(newString.CastToGenericValue());
        }          catch(Exception ex)       {                   ex.log();          }
            try { "CastToHandleValue   : {0}".info(newString.CastToHandleValue());
        }          catch(Exception ex)       {                   ex.log();          }
            try { "CastToHeapValue           : {0}".info(newString.CastToHeapValue());
             }           catch(Exception ex)       {                   ex.log();          }
            try { "CastToObjectValue   : {0}".info(newString.CastToObjectValue());
        }          catch(Exception ex)       {                   ex.log();          }
            try { "CastToReferenceValue      : {0}".info(newString.CastToReferenceValue());
             }           catch(Exception ex)       {                   ex.log();          }
            try { "CastToStringValue   : {0}".info(newString.CastToStringValue());
        }          catch(Exception ex)       {                   ex.log();          }
```

Where only CastToHandleValue **and** CastToReferenceValue **dont return null or throw an exception**

[2:29:13 PM] ERROR: Unable to cast COM object of type 'System.__ComObject' to interface type 'Microsoft.Samples.Debugging.CorDebug.NativeApi.ICorDebugStringValue'. This operation failed because the QueryInterface call on the COM component for the interface with IID '{CC7BCAFD-8A68-11D2-983C-0000F808342D}' failed due to the following error: No such interface supported (Exception from HRESULT: 0x80004002 (E_NOINTERFACE)).
[2:29:13 PM] INFO: CastToReferenceValue : Microsoft.Samples.Debugging.CorDebug.CorReferenceValue
[2:29:13 PM] ERROR: Unable to cast COM object of type 'System.__ComObject' to interface type 'Microsoft.Samples.Debugging.CorDebug.NativeApi.ICorDebugObjectValue'. This operation failed because the QueryInterface call on the COM component for the interface with IID '{18AD3D6E-B7D2-11D2-BD04-0000F80849BD}' failed due to the following error: No such interface supported (Exception from HRESULT: 0x80004002 (E_NOINTERFACE)).

```
[2:29:13 PM] INFO: CastToHeapValue                :
[2:29:13 PM] INFO: CastToHandleValue          : Microsoft.Samples.Debugging.CorDebug.CorHandleValue
[2:29:13 PM] INFO: CastToGenericValue         :
[2:29:13 PM] INFO: CastToBoxValue             :
[2:29:13 PM] INFO: CastToArrayValue                :
```

And I seem to get a pointer to it

```
var newString = debugger.create_String("a new string  ");
return newString.CastToReferenceValue().Value;
```

Ok, lets try via the CreateValue method (the strings above were created using the NewString method)

```
var corEval = debugger.corEval();
char charValue = 'C';
var charType = debugger.process().ResolveClass(charValue.typeFullName());
CorValue val = corEval.CreateValue(CorElementType.ELEMENT_TYPE_CHAR , charType);
var genericValue = val.CastToGenericValue();//
genericValue.SetValue(charValue);
return ((char)genericValue.GetValue()).str();
```

Or more simply

```
var genericValue = debugger.create_Object(CorElementType.ELEMENT_TYPE_CHAR, 'C');
genericValue.value('A');
return ((char)genericValue.GetValue()).str();
```

Or more simply

```
var genericValue = debugger.create_Char('F');
//return ((char)genericValue.GetValue()).str();

typeof(Console).method_bySignature("Void WriteLine(Char)")
                        .invoke_Method(debugger, new CorValue[] { genericValue});
```

Setting a bool

```
var genericValue = debugger.create_Object(CorElementType.ELEMENT_TYPE_BOOLEAN, false);

//return ((bool)genericValue.GetValue()).str();

typeof(Console).method_bySignature("Void WriteLine(Boolean)")
                .invoke_Method(debugger, new CorValue[] { genericValue});
```

or just

```
var genericValue = debugger.create_Bool(true);

typeof(Console).method_bySignature("Void WriteLine(Boolean)")
                        .invoke_Method(debugger, new CorValue[] { genericValue});
```

It doesn't seem that we can create strings this way:

```
var corClass = debugger.process().ResolveClass("a".typeFullName());
var _corValue = corEval.CreateValue(CorElementType.ELEMENT_TYPE_STRING , corClass);
```

throws:

```
InnerException:    at Microsoft.Samples.Debugging.CorDebug.NativeApi.ICorDebugEval.CreateValue(CorElementType elementType, ICorDebugClass
pElementClass, ICorDebugValue& ppValue)
   at Microsoft.Samples.Debugging.CorDebug.CorEval.CreateValue(CorElementType type, CorClass managedClass) in e:\O2_V4\_O2_V4_TempDir
\_ToolsOrAPIs\MDbg_Sample\MDbg Sample\src\debugger\corapi\Eval.cs:line 199
```

Recaping what we have so far:

Start a process, write console message and stop it after 5 seconds:

```
MDbgEngine debugger = new MDbgEngine();
var process = debugger.CreateProcess(exeToDebug, "", DebugModeFlag.Debug, null);
```

```
debugger.process_StopInNSeconds(5);
debugger.waitOne();
typeof(Console).method_bySignature("Void WriteLine(System.String)")
                        .invoke_Method(debugger, "Hello from the debugger");
```

or more simply

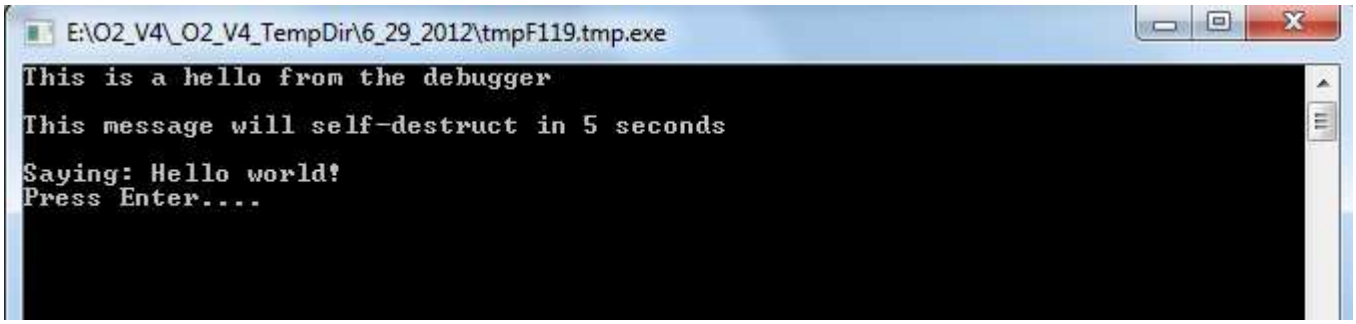
```
MDbgEngine debugger = new MDbgEngine();
var process = debugger.CreateProcess(exeToDebug, "", DebugModeFlag.Debug, null);
debugger.process_StopInNSeconds(5);
debugger.waitOne();
debugger.console_WriteLine("This is a hello from the debugger");
```

or more simply:

```
MDbgEngine debugger = new MDbgEngine();
var process = debugger.CreateProcess(exeToDebug, "", DebugModeFlag.Debug, null);
debugger.process_StopInNSeconds(5)
        .waitOne()
        .console_WriteLine("This is a hello from the debugger")
        .console_WriteLine()
        .console_WriteLine("This message will self-destruct in 5 seconds \n")
        .go();
```

or more simply

```
new MDbgEngine().startProcess(exeToDebug)
                .process_StopInNSeconds(5)
                .waitOne()
                .console_WriteLine("This is a hello from the debugger")
                .console_WriteLine()
                .console_WriteLine("This message will self-destruct in 5 seconds \n")
                .go();
```



We can also start the process, write the message and detach (leaving the process to execute)

```
new MDbgEngine().startProcess(exeToDebug)
                .waitOne()
                .console_WriteLine("This is a hello from the debugger")
                .console_WriteLine()
                .console_WriteLine("This message will self-destruct in 5 seconds \n")
                .detach();
```

Here is how to get the list of avaiable process to attach:

```
        public static List<Process> attachableProcesses(this MDbgEngine engine)
        {
                var attachableProcesses = new List<Process>();
                        foreach (var process in Process.GetProcesses())
                  {
                     if (Process.GetCurrentProcess().Id == process.Id)  // let's hide our process
                         continue;

                     CLRMetaHost mh = null;
                     try
                     {
                         mh = new CLRMetaHost();
```

```
                }
                catch (System.EntryPointNotFoundException)
                {
                    continue;
                }

                IEnumerable<CLRRuntimeInfo> runtimes = null;
                try
                {
                    runtimes = mh.EnumerateLoadedRuntimes(process.Id);
                }
                catch
                {
                    continue;
                }

                //if there are no runtimes in the target process, don't print it out
                if (!runtimes.GetEnumerator().MoveNext())
                    continue;
                attachableProcesses.add(process);
            }
            return attachableProcesses;
            }
        }
```

which can be consumed like this:

```
var attachableProcesses = debugger.attachableProcesses();

return attachableProcesses;
```

It is usefull to see the ids and version of these processes:

```
        public static string clrDetails(this List<Process> clrProcesses)
            {
                var clrDetails = "";
                foreach(var clrProcess in clrProcesses)
                {
                    var id = clrProcess.Id;
                    var name = clrProcess.ProcessName;
                    var fileName =  clrProcess.MainModule.FileName;
                    var runtimes = new CLRMetaHost().EnumerateLoadedRuntimes(id);
                    var versions = (from runtime in runtimes
                                                select runtime.GetVersionString()).toList().join
(",");
                    clrDetails += "id: {0}  name: {1}  runtimes: {2} fileName: {3}".line().format
(id, name, versions, fileName);
                }
                return clrDetails;
            }
```

like this:

```
return debugger.attachableProcesses().clrDetails();
```

Output

```
id: 8784   name: Cropper       runtimes: v2.0.50727        fileName: E:\O2_V4\_O2_V4_TempDir\_ReferencesDownloaded\Cropper.exe
id: 20228  name: tmp4AAC.tmp   runtimes: v4.0.30319        fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\tmp4AAC.tmp.exe
id: 31264  name: tmp576E.tmp   runtimes: v4.0.30319        fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\tmp576E.tmp.exe
id: 56068  name: tmp7B6C.tmp   runtimes: v4.0.30319        fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\tmp7B6C.tmp.exe
id: 9960   name: Sample Exe - 9506   runtimes: v4.0.30319  fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\_tempExes\Sample Exe - 9506.exe
id: 17612  name: Sample Exe - 8529   runtimes: v4.0.30319  fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\_tempExes\Sample Exe - 8529.exe
id: 9324   name: tmpEDB5.tmp   runtimes: v4.0.30319        fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\tmpEDB5.tmp.exe
id: 63468  name: tmpA6E9.tmp   runtimes: v4.0.30319        fileName: E:\O2_V4\_O2_V4_TempDir\6_29_2012\tmpA6E9.tmp.exe
id: 1796   name: O2 Platform - 4.0 x64   runtimes: v4.0.30319  fileName: E:\O2_V4\O2.Platform.Projects\binaries\O2 Platform - 4.0 x64.exe
id: 3952   name: ILSpy        runtimes: v4.0.30319        fileName: E:\_Tests\ILSpy\ILSpy\bin\Debug\ILSpy.exe
id: 9660   name: Cropper       runtimes: v2.0.50727        fileName: E:\O2_V4\_O2_V4_TempDir\_ReferencesDownloaded\Cropper.exe
```

you can get the process to attach by name:

```
return debugger.attachableProcesses().with_Name("tmp576E.tmp") ;
```

or by id:

```
return debugger.attachableProcesses().with_Id(31264) ;
```

.

.

.

.

..

Good References:

- Mdbg watch-trace extension http://lowleveldesign.wordpress.com/2012/02/27/mdbg-watch-trace-extension
- Using Managed Code to debug Memory Dumps **http://naveensrinivasan.com/2010/11/11/using-managed-code-to-debug-memory-dumps/**
- http://naveensrinivasan.com/category/windbg/page/2/ - good examples of WinBbg scripting (like for loops)
- .