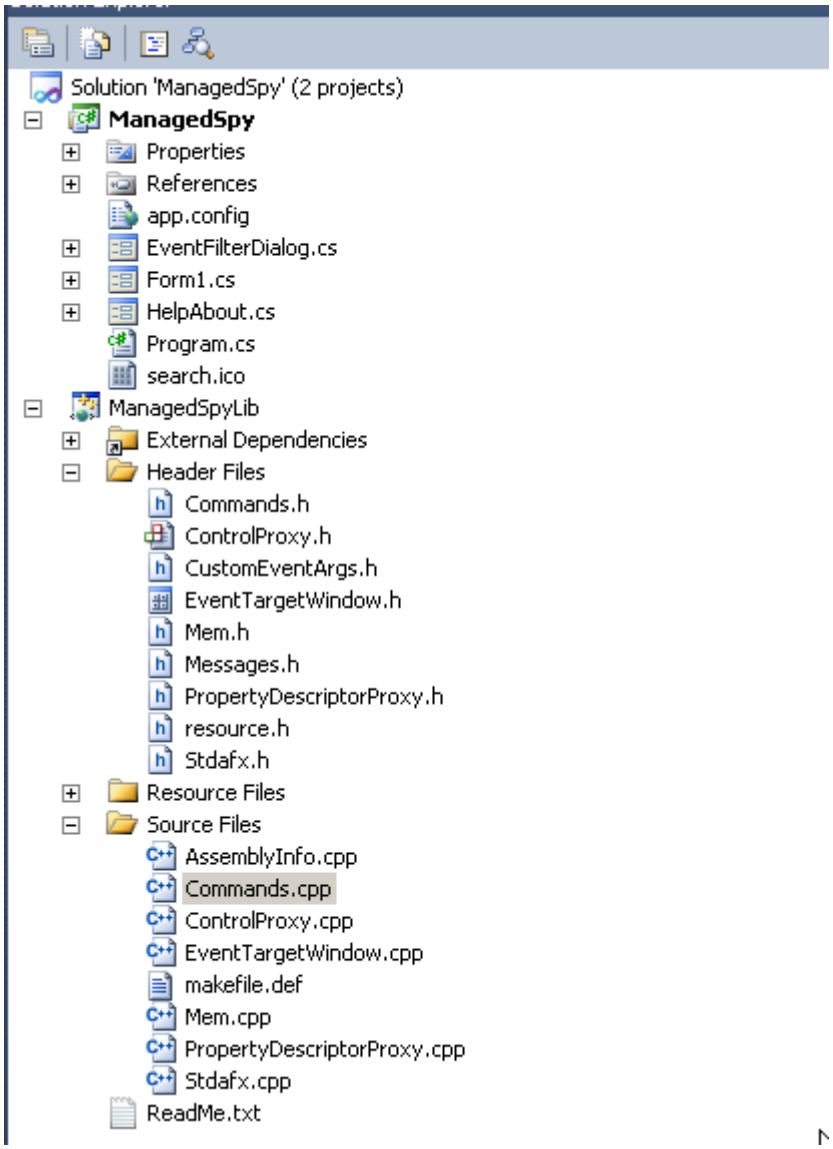
ManagedSpy - Using and coding the ManagedSpyLib APIs

based on the article http://msdn.microsoft.com/en-us/magazine/cc163617.aspx and the code http://download.microsoft.com/download/f/2/7/f279e71e-efb0-4155-873d-5554a0608523/ManagedSpy.exe

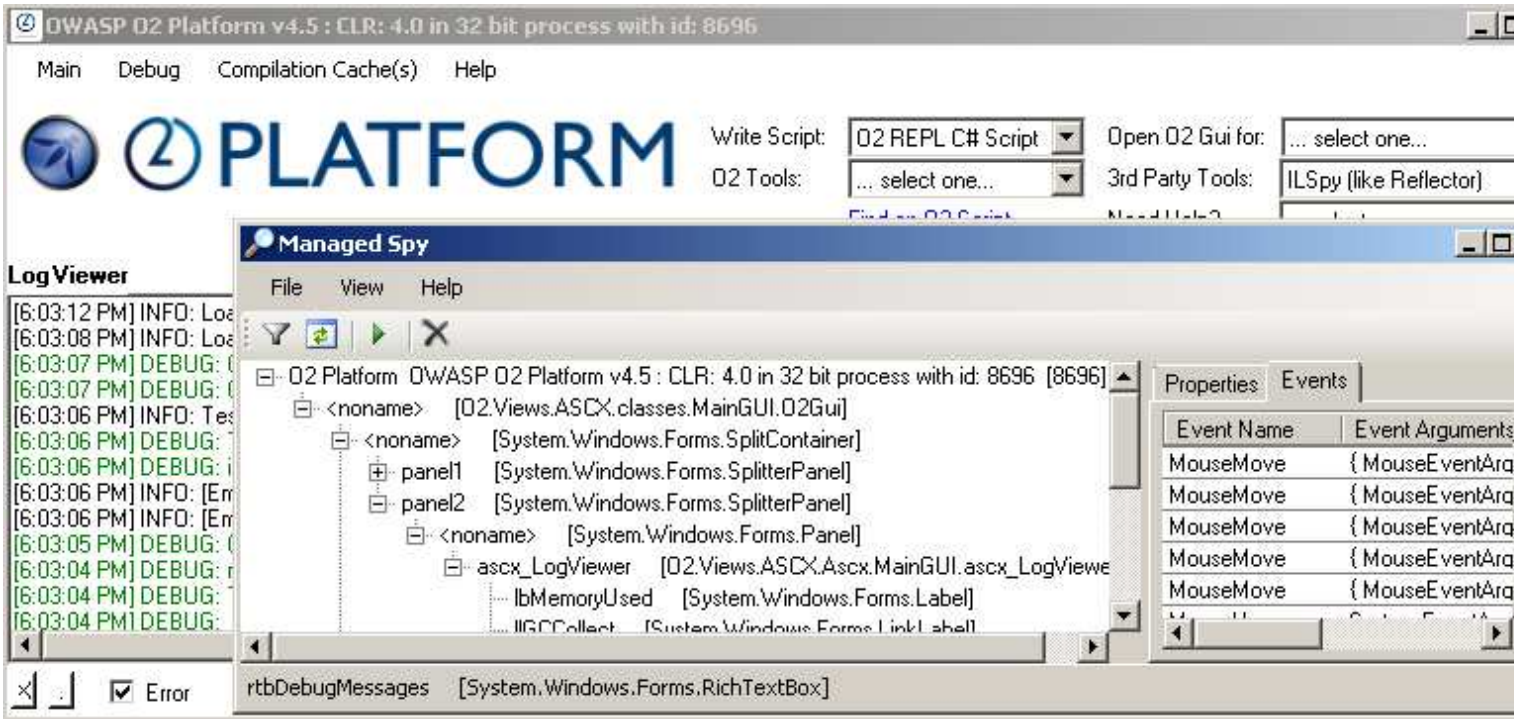
I unzipped the code, opened it up on VS2010 (which needed an upgrade):



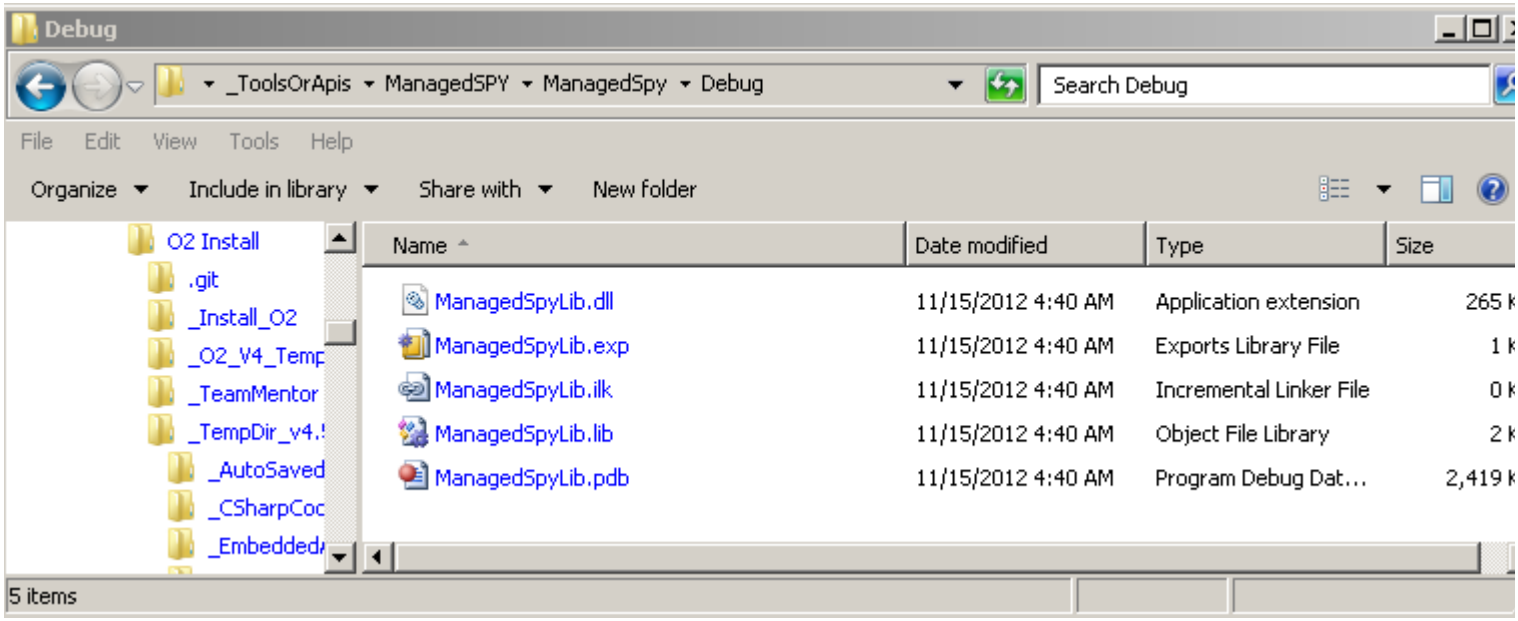
with a couple code changes to allow it to run on 4.0

```
Commands.cpp | Stdafx.cpp | PropertyDescriptorProxy.cpp | Mem.cpp | ControlProxy.cpp | ControlProxy.h | ManagedSpy | ReadMe.tx
Desktop                                                  IsManagedProcess(DWORD processID)
    if (proc == nullptr) {
        return false;
    }

    bool ismanaged = false;
    try
    {
        for(int i = 0;i<proc->Modules->Count;i++) {
            if(proc->Modules[i]->ModuleName == _T("mscorlib.dll") ||
                proc->Modules[i]->ModuleName == _T("mscorlib.ni.dll")) {
                    //dc
                    //make sure its version 2.0
                //  System::Reflection::AssemblyName^ name = System::Reflection::AssemblyName::GetAssemblyNam
                    //  proc->Modules[i]->FileName);
            //      if (name != nullptr && name->Version->Major == 2) {
                        ismanaged = true;
                    //}
                break;
            }
        }
    }
```

And here it is in action, detecting some windows messages on a .net 4.0 process

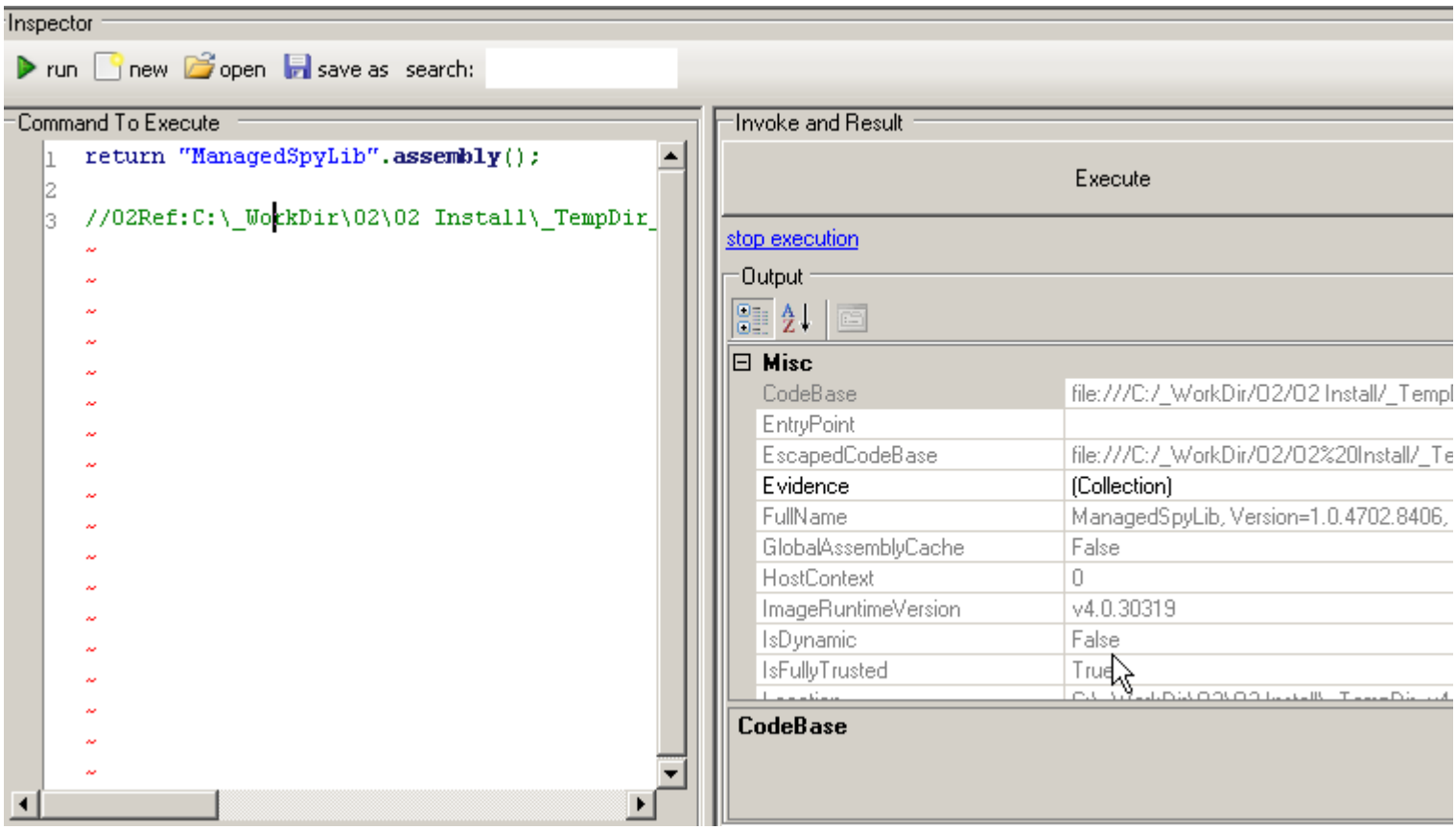


Next we want to use the powerful ManagedSpyLib.dll

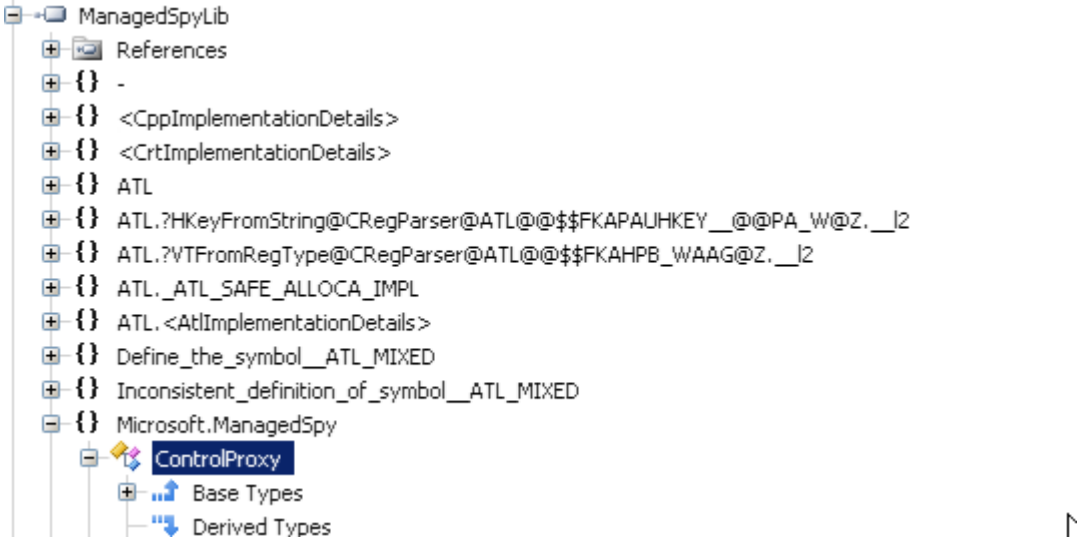


Which we can consume directly on the O2 C# REPL environment:

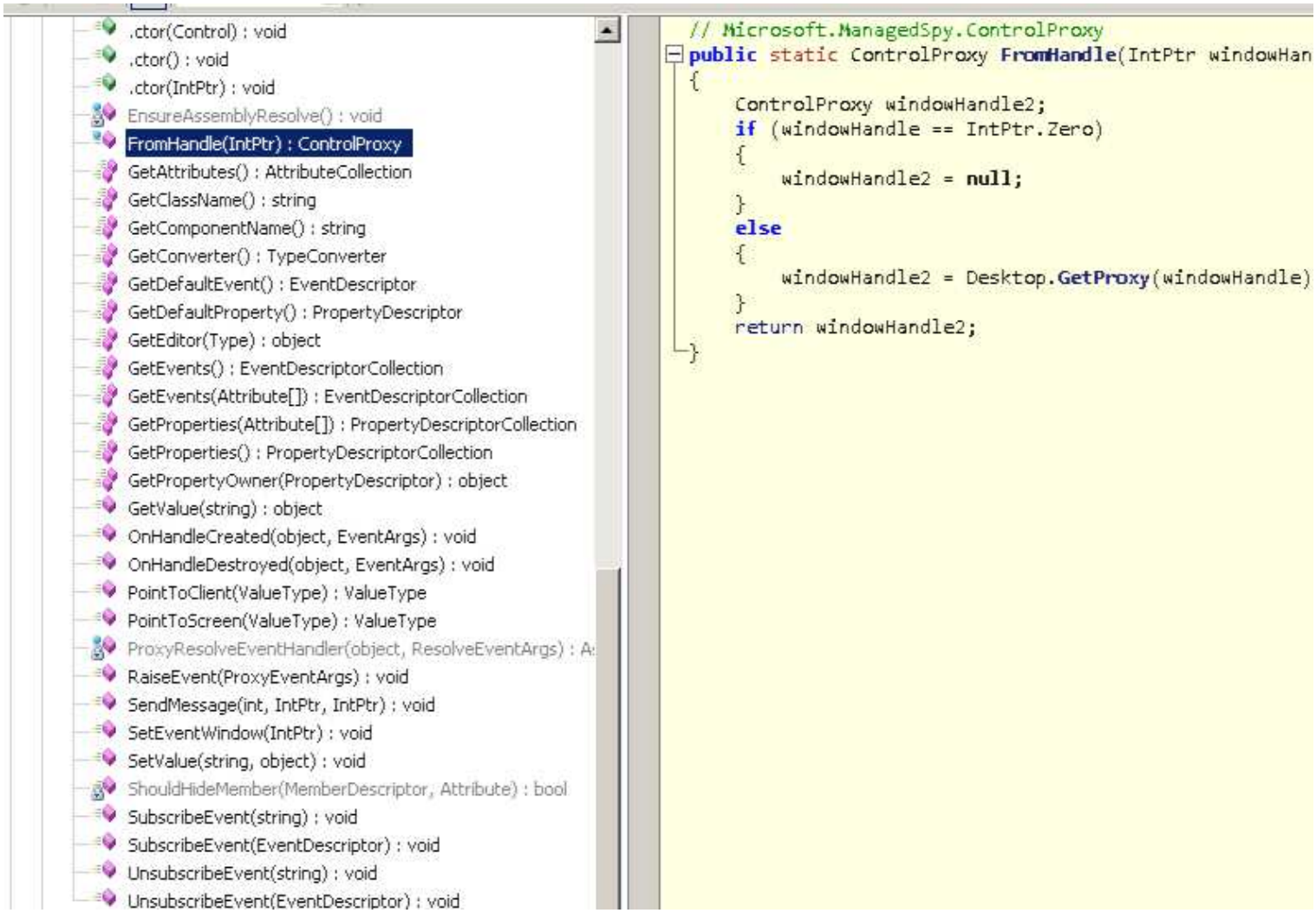
```
return "ManagedSpyLib".assembly();

//O2Ref:C:\_WorkDir\O2\O2 Install\_TempDir_v4.5.1.0\_ToolsOrApis\ManagedSPY\ManagedSpy\Debug
\ManagedSpyLib.dll
```



The class we want to use is the ControlProxy



Namely the methods that allow access to the target process via reflection:

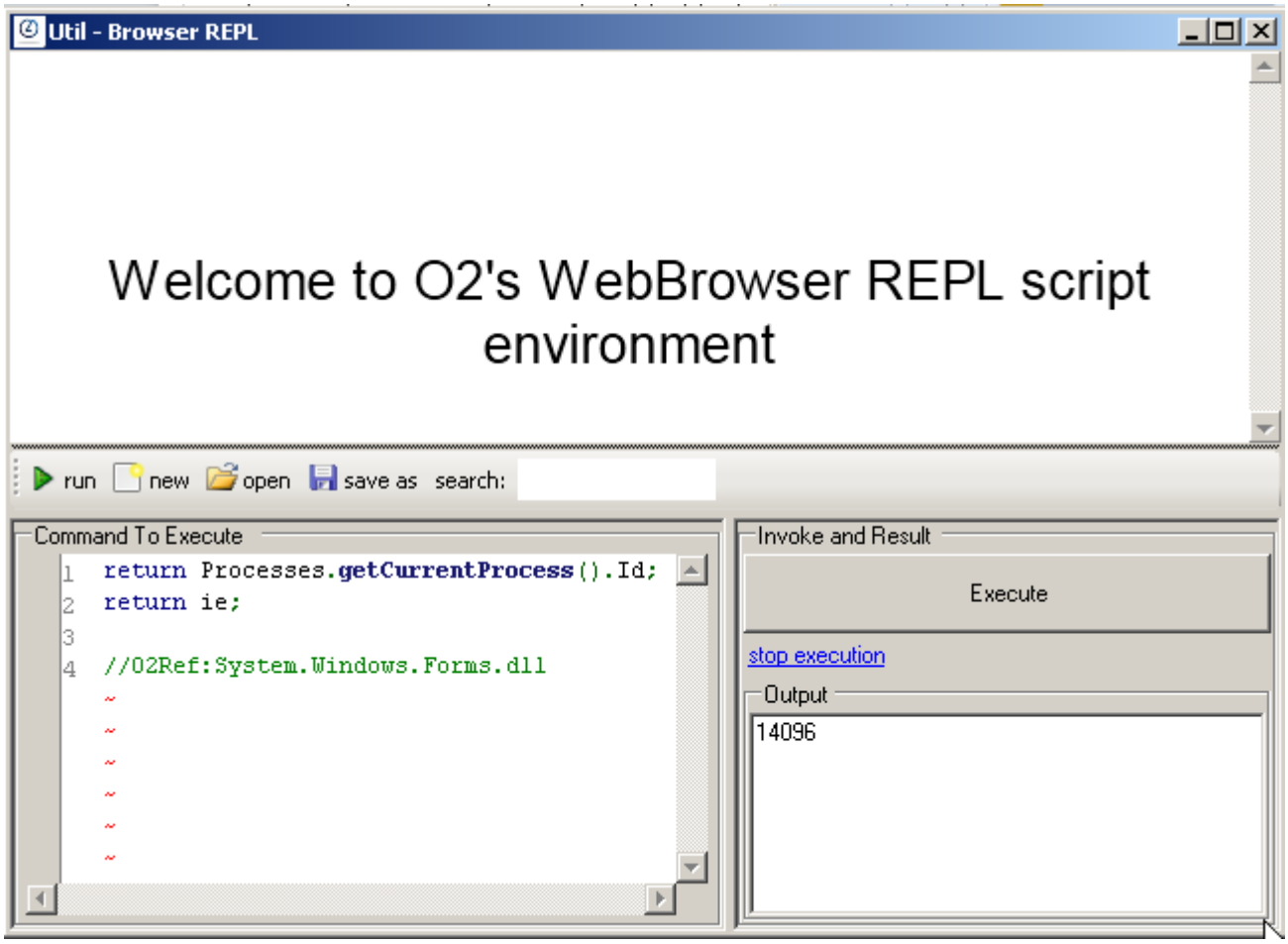


Here is an example from the original MSDN article:

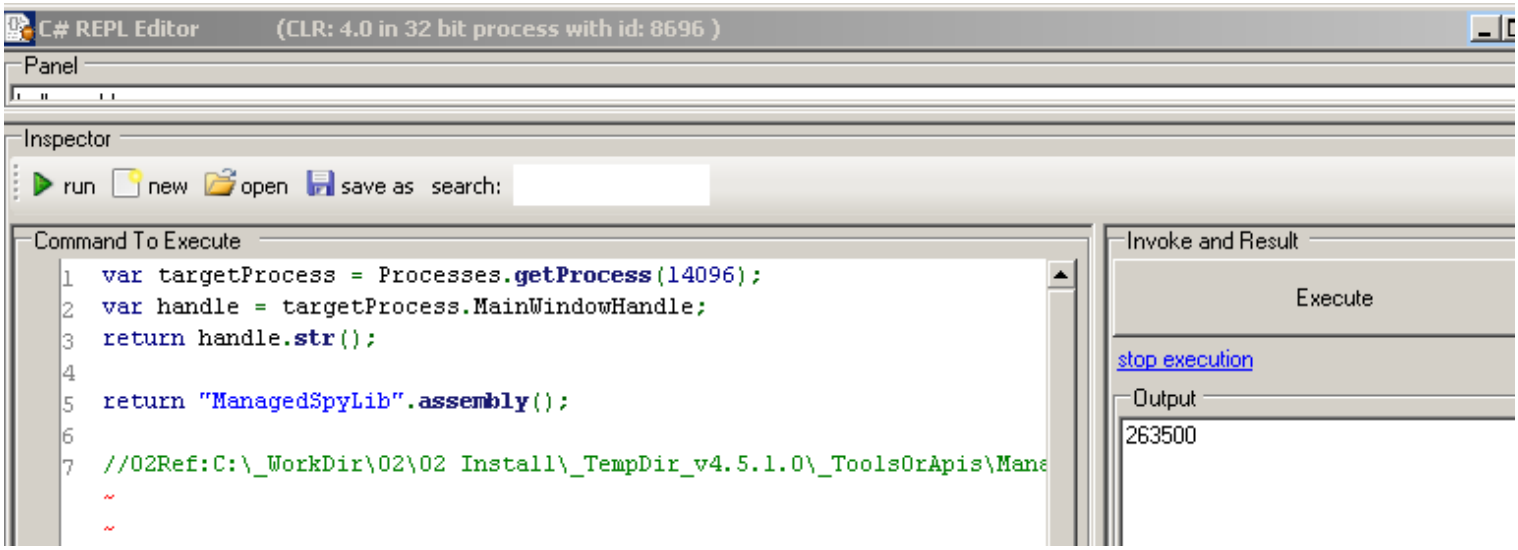
Figure 11 Testing Code

```
private void button1_Click(object sender, EventArgs e)
{
    Process[] procs = Process.GetProcessesByName("Multiply");
    if (procs.Length != 1) return;
    ControlProxy proxy =
        ControlProxy.FromHandle(procs[0].MainWindowHandle);
    if (proxy == null) return;

    //find the controls we are interested in...
    if (cbutton1 == null)
    {
        foreach (ControlProxy child in proxy.Children)
        {
            if (child.GetComponentName() == "textBox1") {
                textBox1 = child;
            }
            else if (child.GetComponentName() == "textBox2") {
                textBox2 = child;
            }
            else if (child.GetComponentName() == "textBox3") {
                textBox3 = child;
            }
            else if (child.GetComponentName() == "button1") {
                cbutton1 = child;
            }
```

The idea is to apply the same API calls to connect to this o2 process (stand alone web REPL on process with ID 14096)

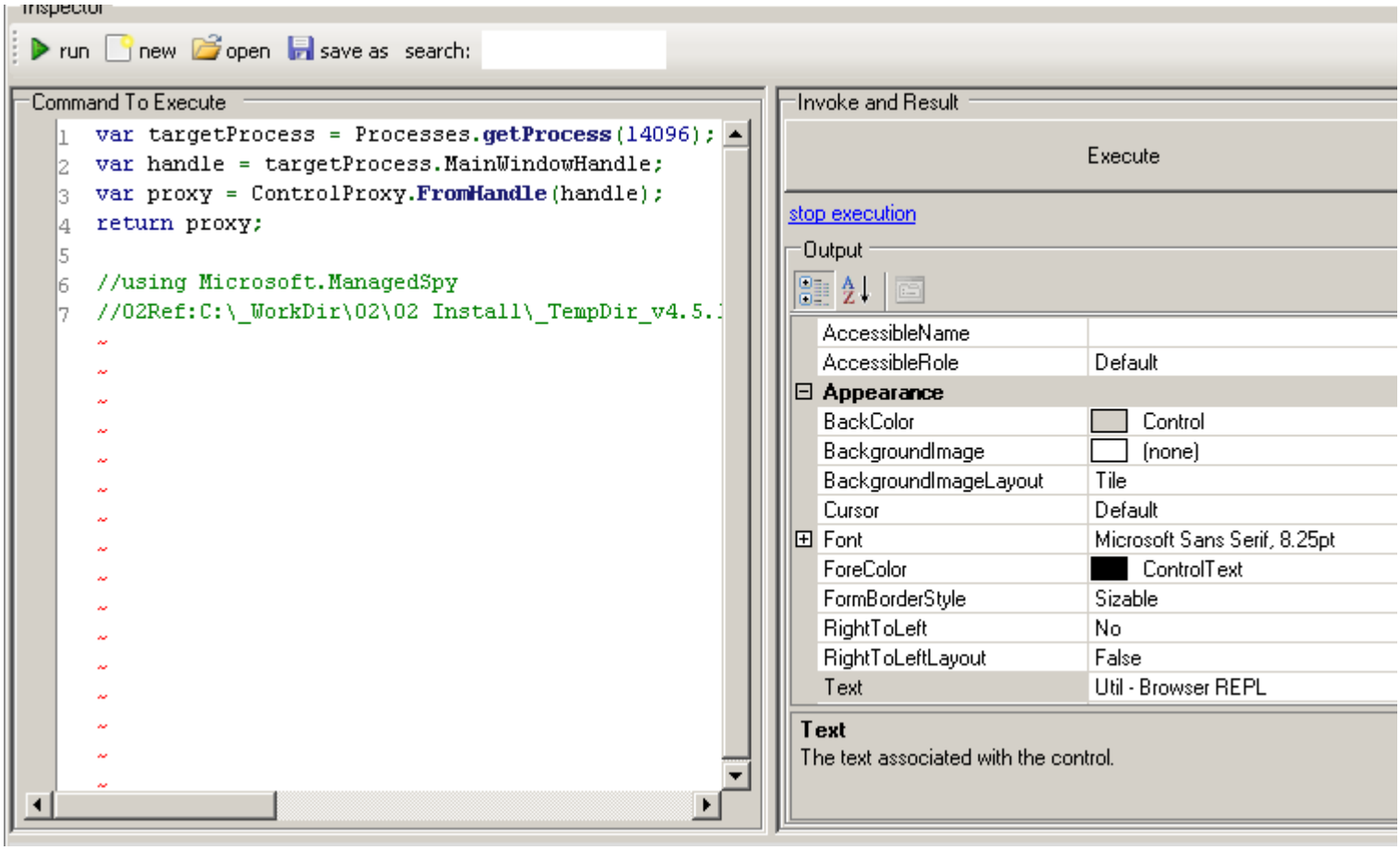


from the C# REPL script environment we loaded the ManagedSpyLib.dll

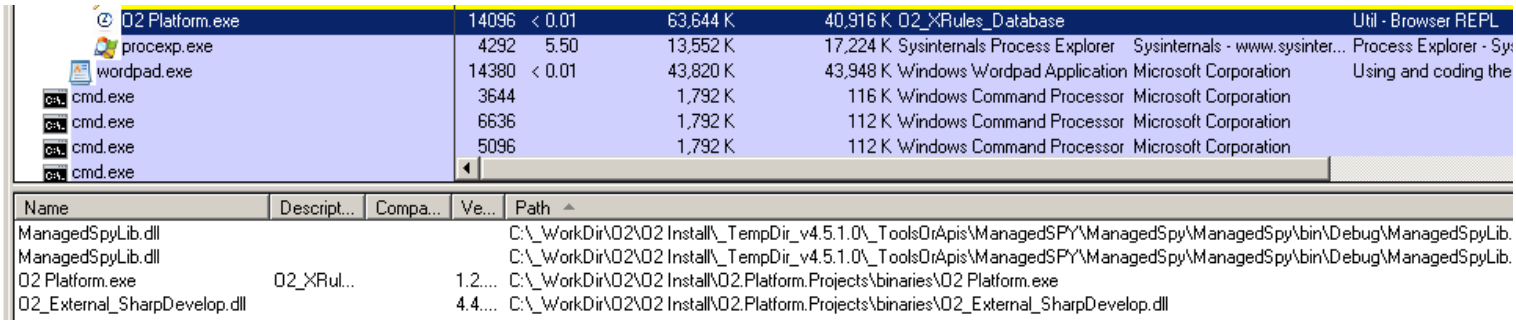


Creating an instance of ControlProxy using the target process' MainWindowHandle

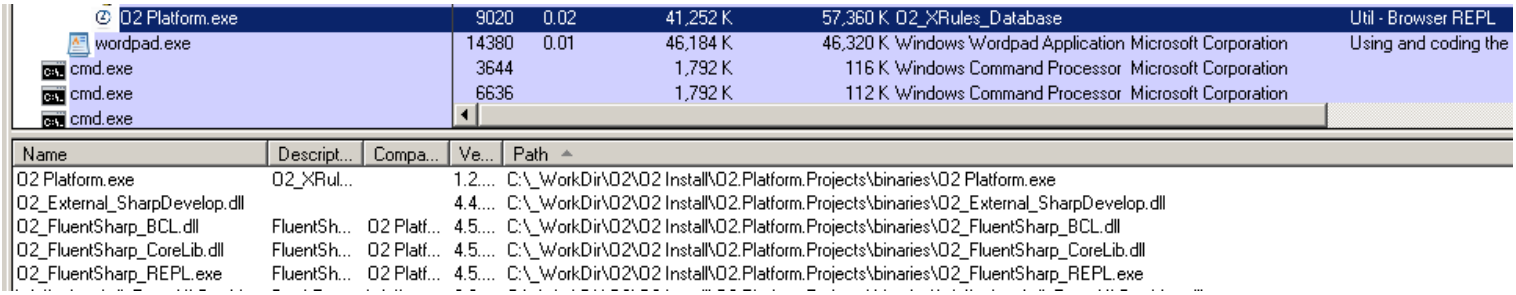
```
var targetProcess = Processes.getProcess(14096);
var handle = targetProcess.MainWindowHandle;
var proxy = ControlProxy.FromHandle(handle);
return proxy;

//using Microsoft.ManagedSpy
//O2Ref:ManagedSpyLib.dll
```



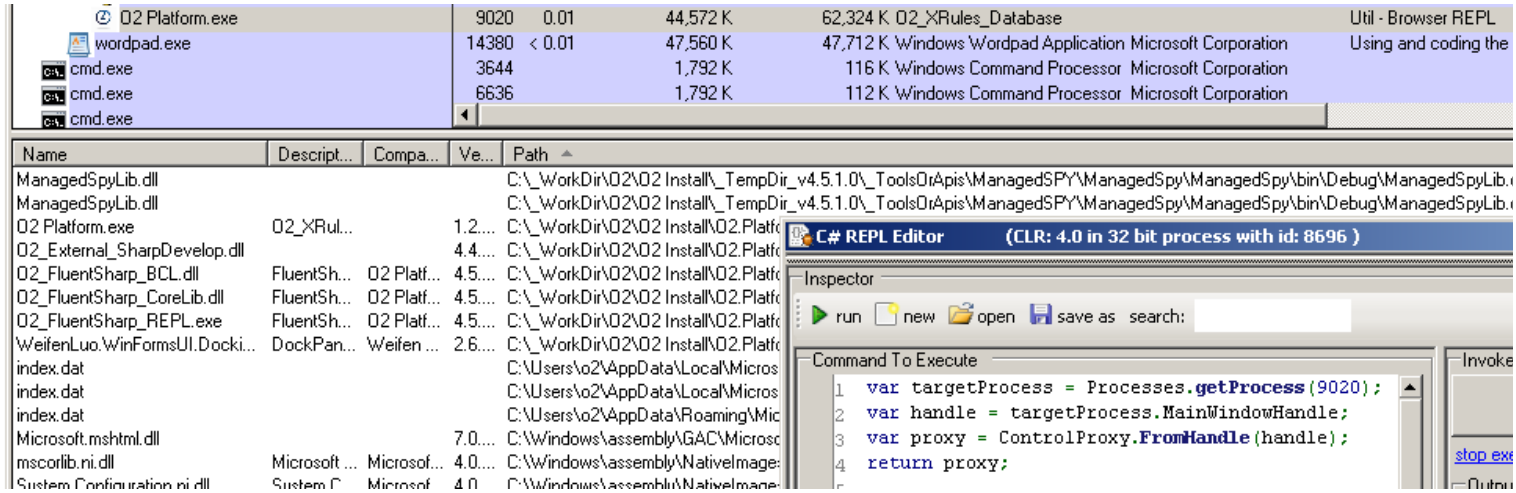
What just happened is that the ManagedSpyLib.dll was just injected into the 14096 process



To double check this, let's open another target process (now with ID 9020) and before the injection the ManagedSpyLib.dll is not there:

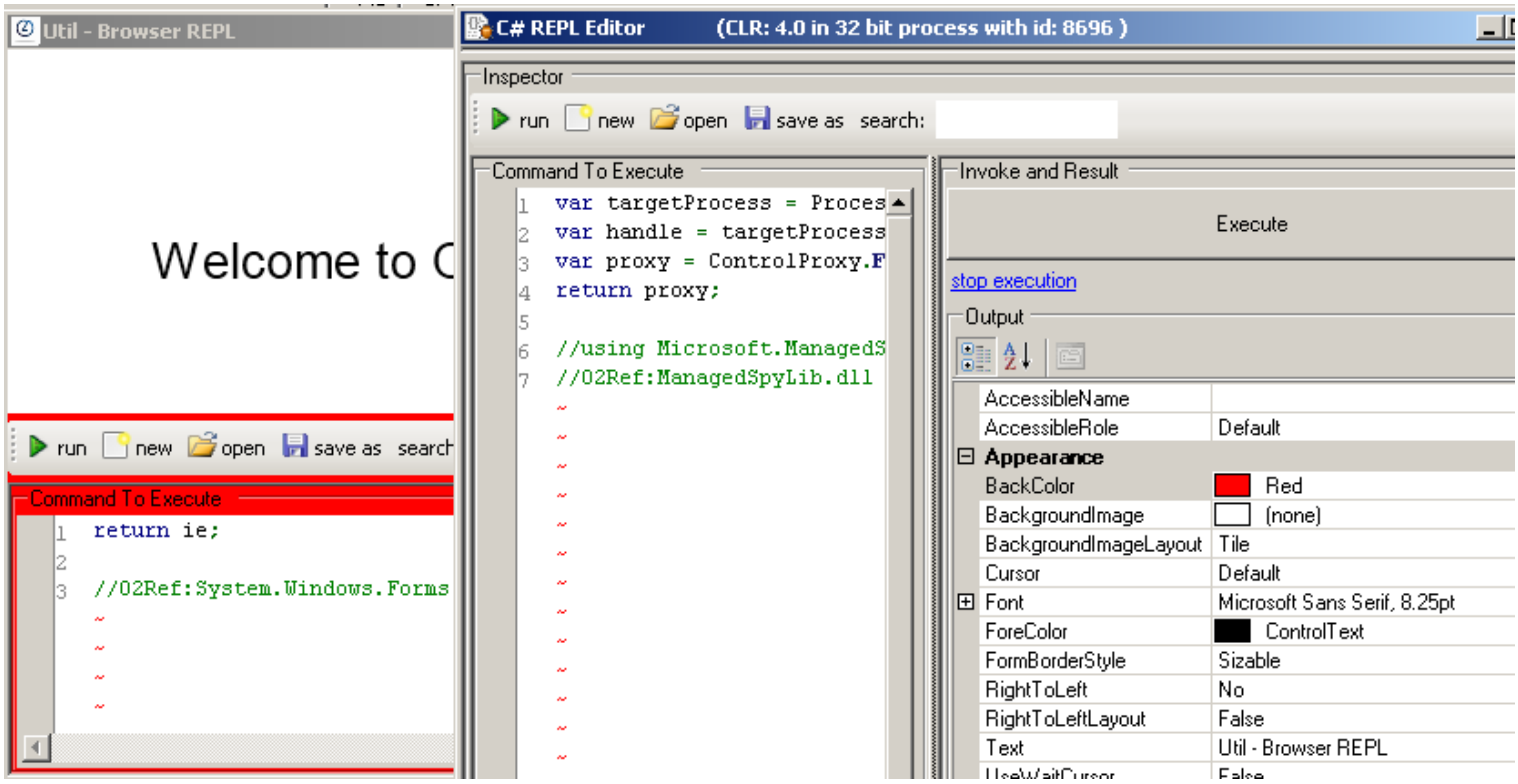


And is there after the injection:

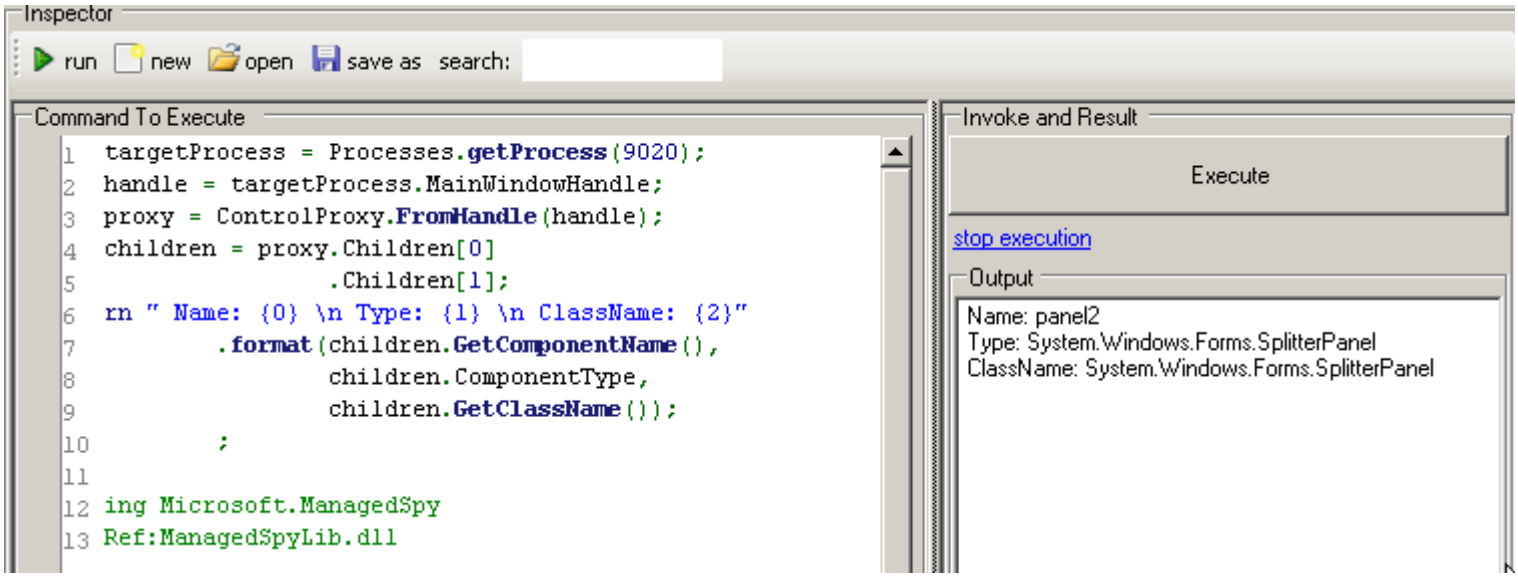


What is already quite interresting is that the ControlProxy fells like a WinForms Control object, and we can change the properties on the remote process which are directly applied to the target process (main window).

In the screenshot below, I changed the back color on the propertyGrid show on the the remote process (right)

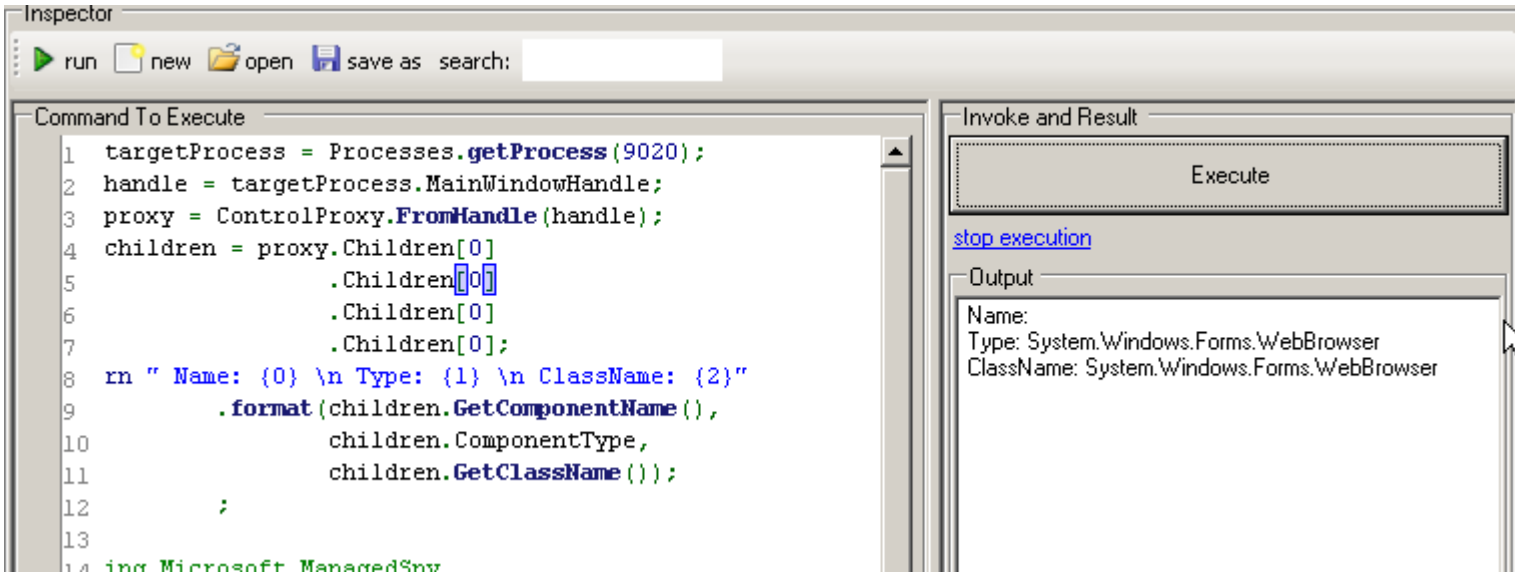


Retrieving information about the current control

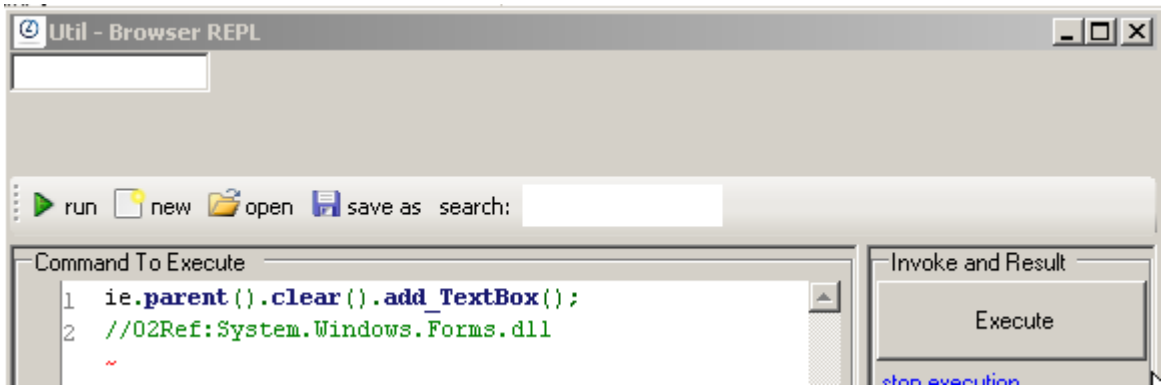


Here is the WebBrowser from the targer process:

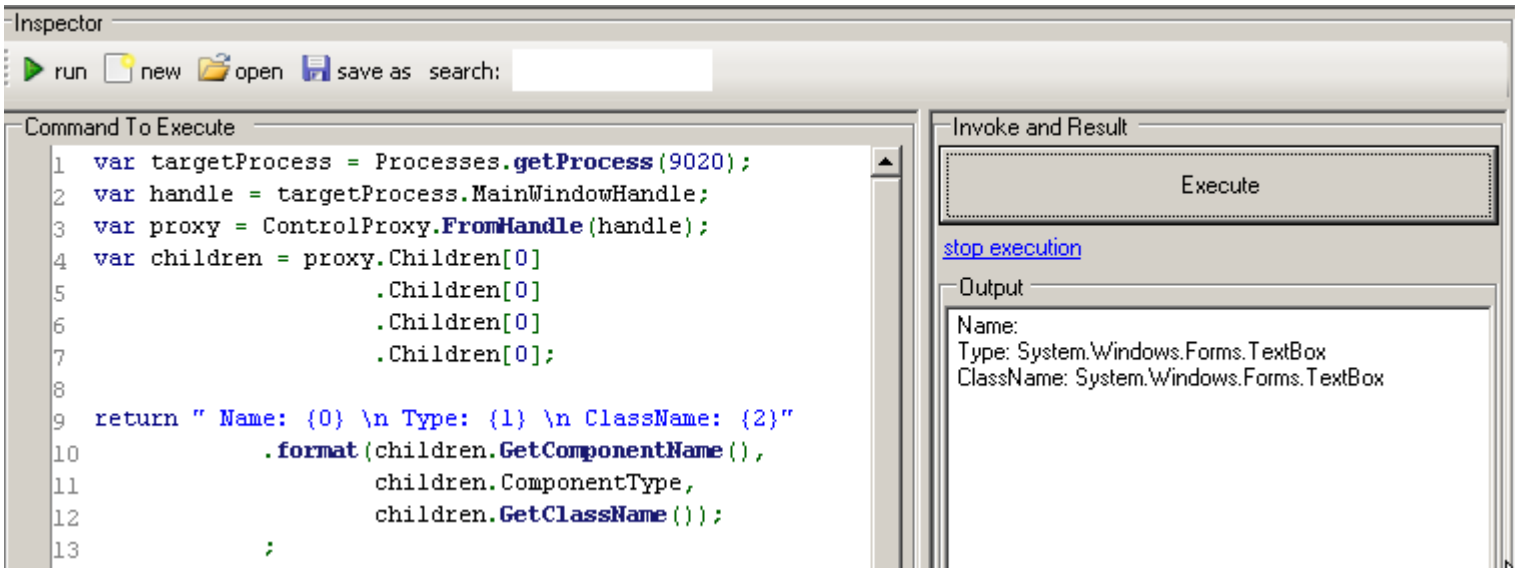
```
var targetProcess = Processes.getProcess(9020);
var handle = targetProcess.MainWindowHandle;
var proxy = ControlProxy.FromHandle(handle);
var children = proxy.Children[0]
                                .Children[0]
                                .Children[0]
                                .Children[0];

return " Name: {0} \n Type: {1} \n ClassName: {2}"
                    .format(children.GetComponentName(),
                                children.ComponentType,
                                children.GetClassName());;

//using Microsoft.ManagedSpy
//O2Ref:ManagedSpyLib.dll
```



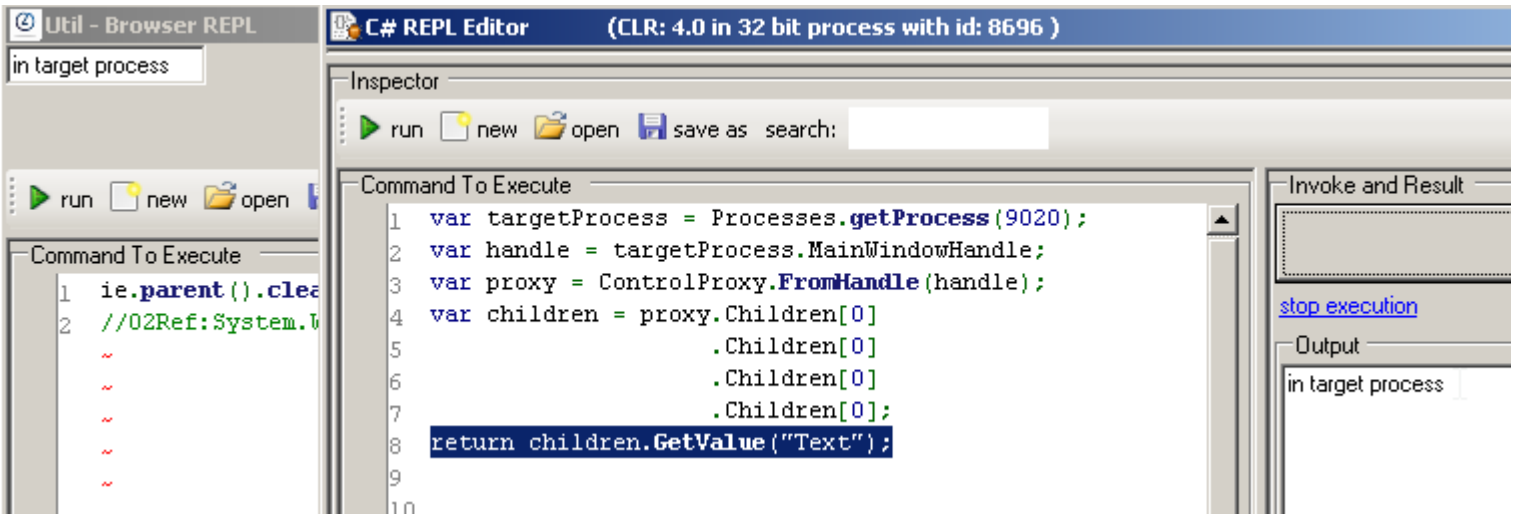
To make it easy on this first example, lets change the WebBrowser into a textbox:



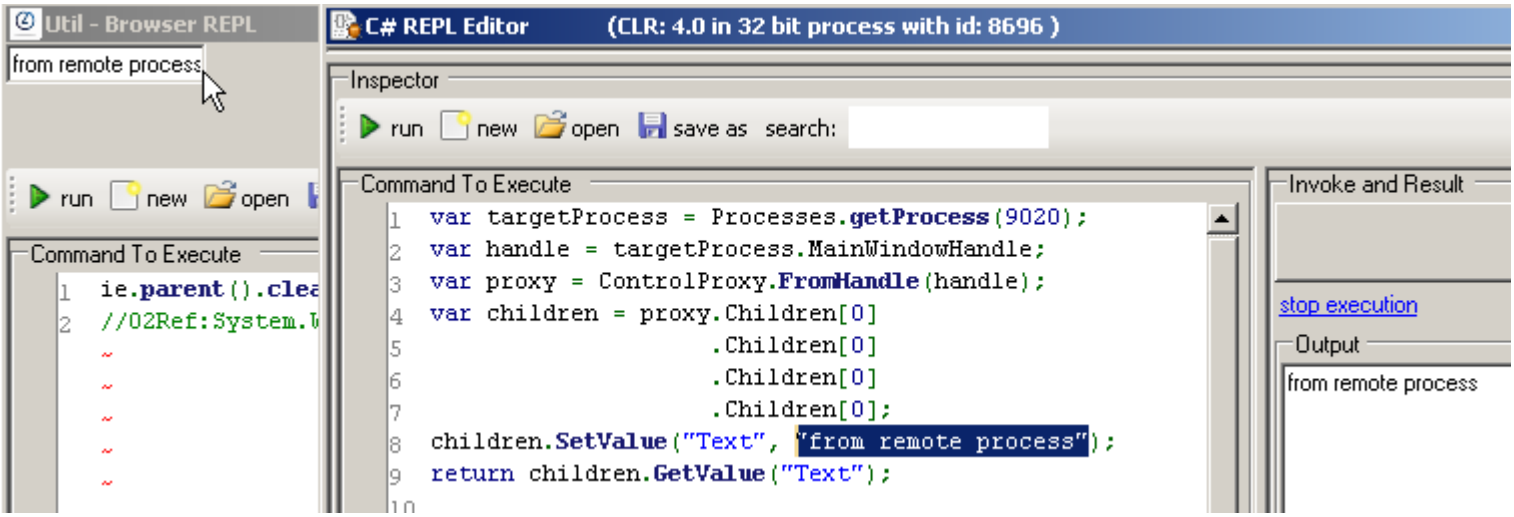
Which is reflected in the remote process:



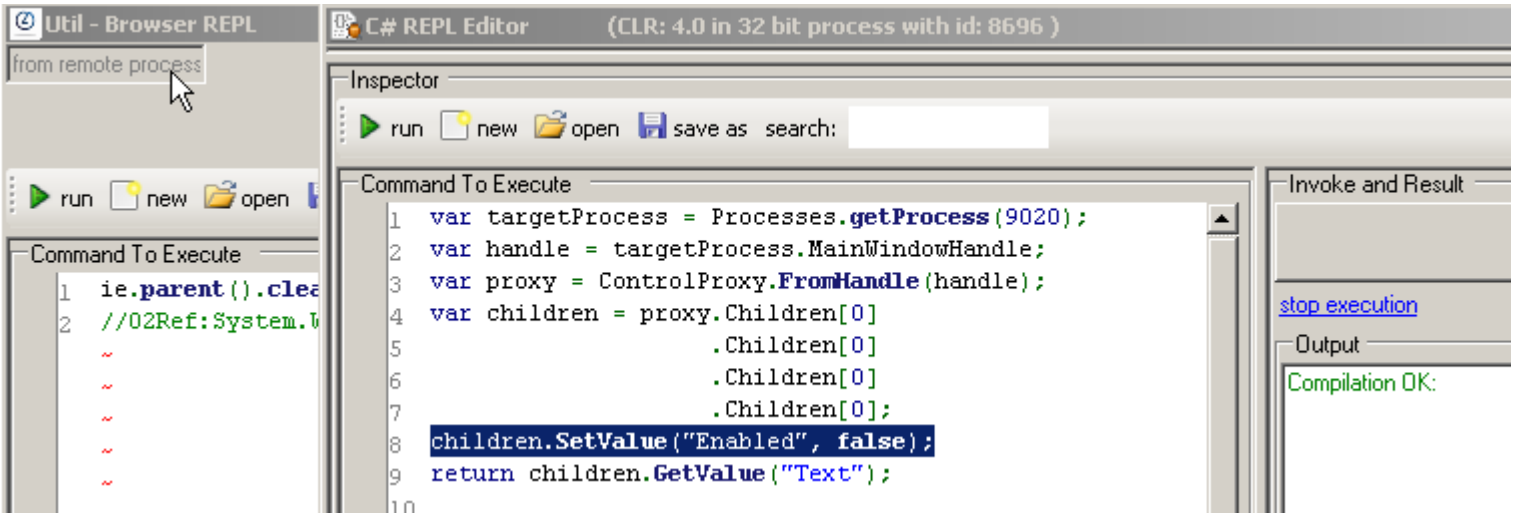
If we change the value of the TextBox we can retrieve it from the remote process:



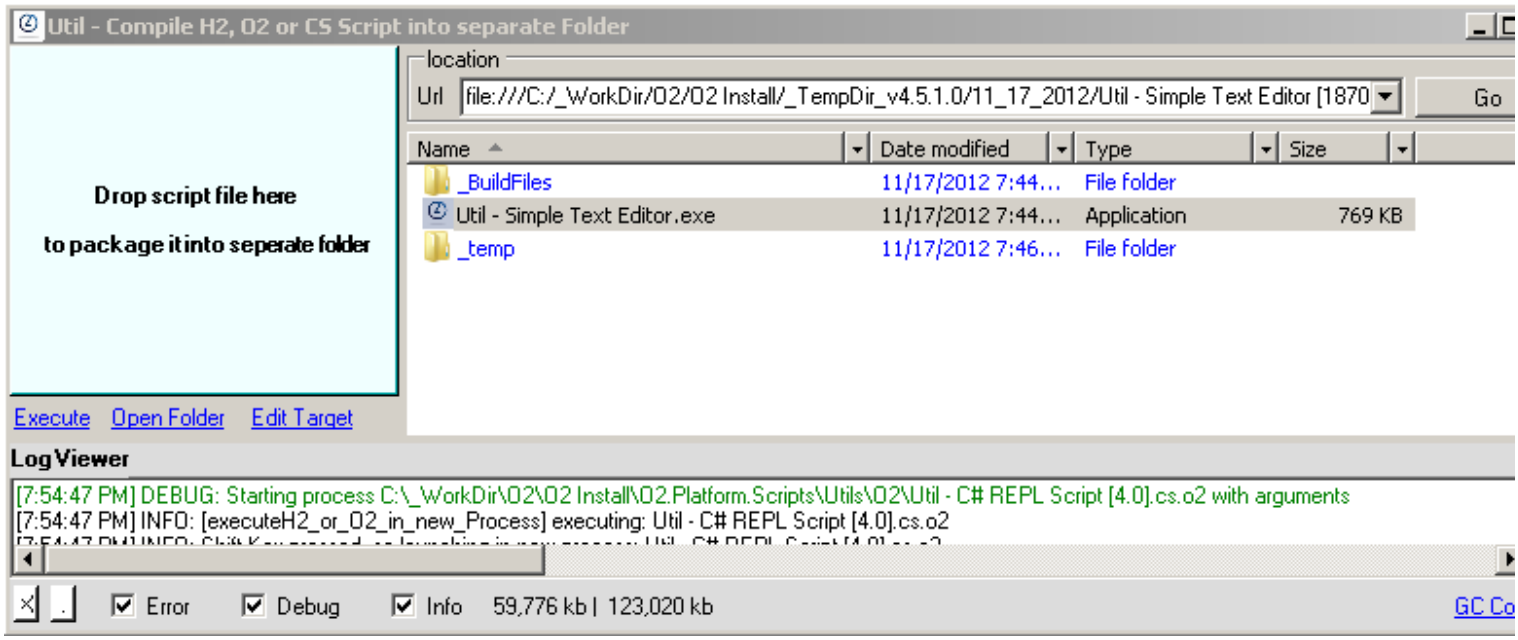
or set it from the remote process:



or invoke other properties like the Enabled



Using a packaged stand alone exe (created from the *Util - Simple Text Editor.h2* script:

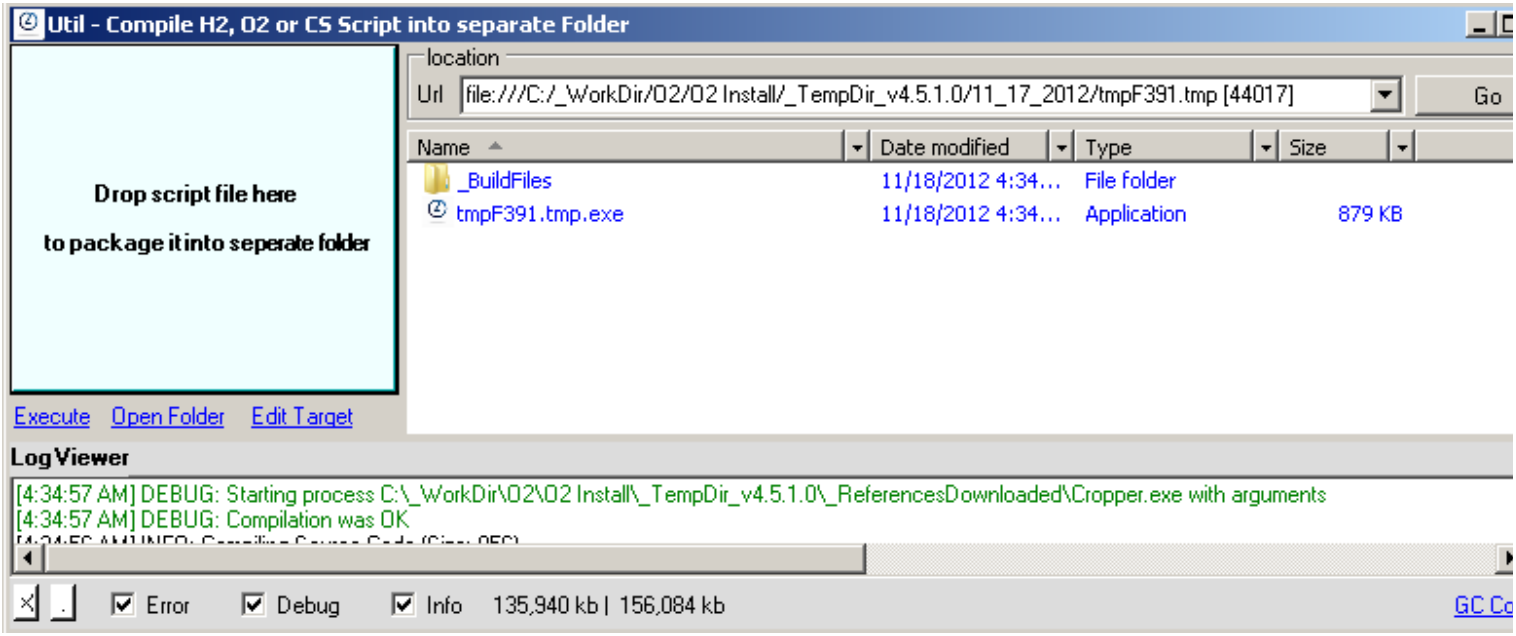


Script to start a .Net process and monitor its events:

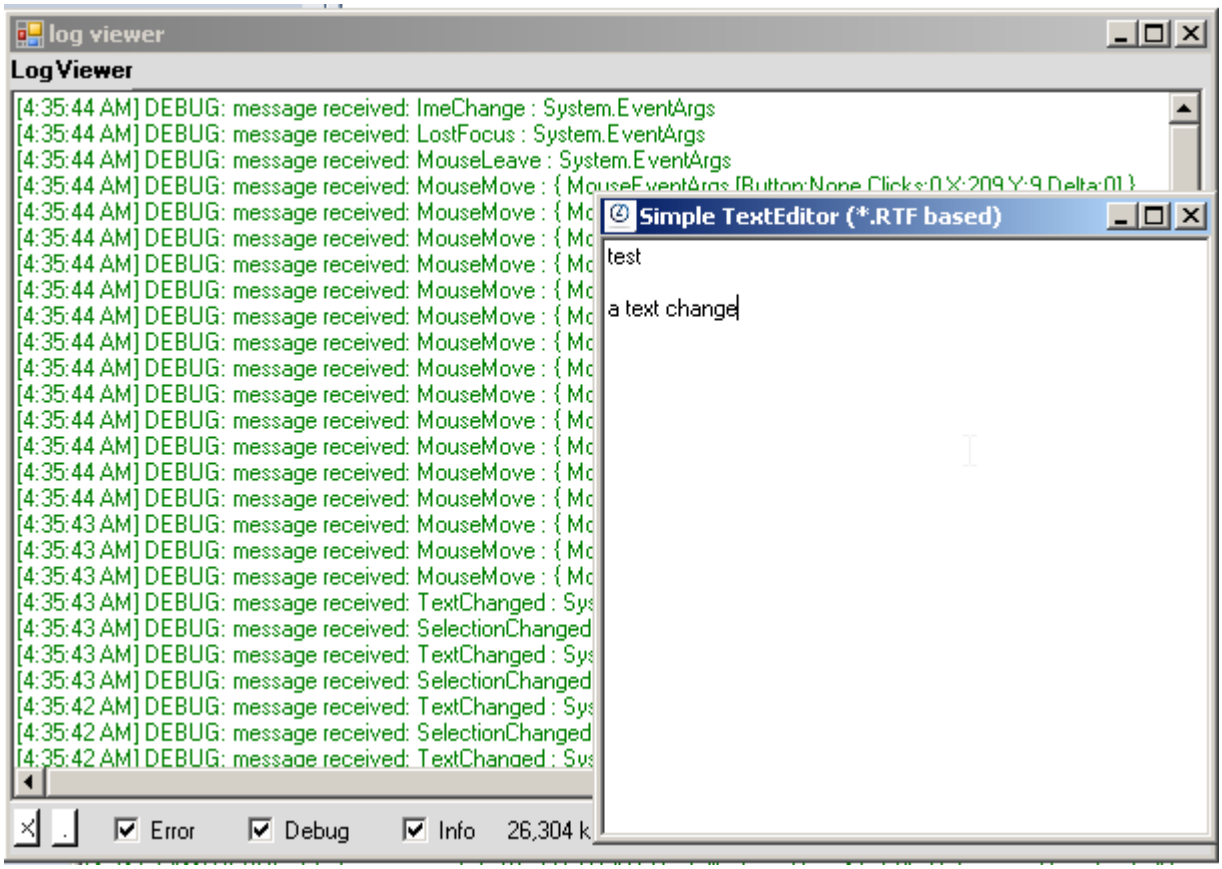
```
"log viewer".popupWindow().add_LogViewer();
var eventWindow = (Control)"ManagedSpyLib.dll".assembly().type("Desktop")
                                                .fieldValue("eventWindow");;

var simpleEditorExe = @"C:/_WorkDir/O2/O2 Install/_TempDir_v4.5.1.0/11_17_2012/Util - Simple Text Editor
[18704]\Util - Simple Text Editor.exe";

var targetProcess =   simpleEditorExe.startProcess();
while(targetProcess.MainWindowHandle == IntPtr.Zero) { targetProcess.sleep(250); }
var handle = targetProcess.MainWindowHandle;
ControlProxy proxy =null;


O2Thread.mtaThread(
       ()=>{
                     eventWindow.invokeOnThread(
                           ()=>{
                                        proxy = ControlProxy.FromHandle(handle);
                                        Application.Run();
                                  });

              });
while(proxy.isNull())
{
       300.sleep();
}
eventWindow.invokeOnThread(
       ()=>{
                     var currentProxy =  proxy.Children[0];//.GetValue("Text");

                     currentProxy.EventFired += new ControlProxyEventHandler(
                                        (sender, args)=>{

                                                                                  var messageDetails =
"{0} : {1}".format(args.eventDescriptor.Name, args.eventArgs.ToString());
                                                                                  "message received:
{0}".debug(messageDetails);
                                                                        });


                     currentProxy.SetValue("Text", "test");
                     foreach (EventDescriptor @event in currentProxy.GetEvents())
                           currentProxy.SubscribeEvent(@event.Name);

                     System.Diagnostics.Debug.WriteLine("Configured ControlProxy");
                     "Events hook setup".info();
              });

targetProcess.WaitForExit();
Application.Exit(null);
```

```
return "done";

//using System.ComponentModel
//using Microsoft.ManagedSpy

//generateDebugSymbols
//O2Ref:ManagedSpyLib.dll
```

Note that this script (above) needs to run under a new process (so we will run it from a 'packaged O2 Script')



here is the script in action



Improved version

```
"log viewer".popupWindow().add_LogViewer();
```

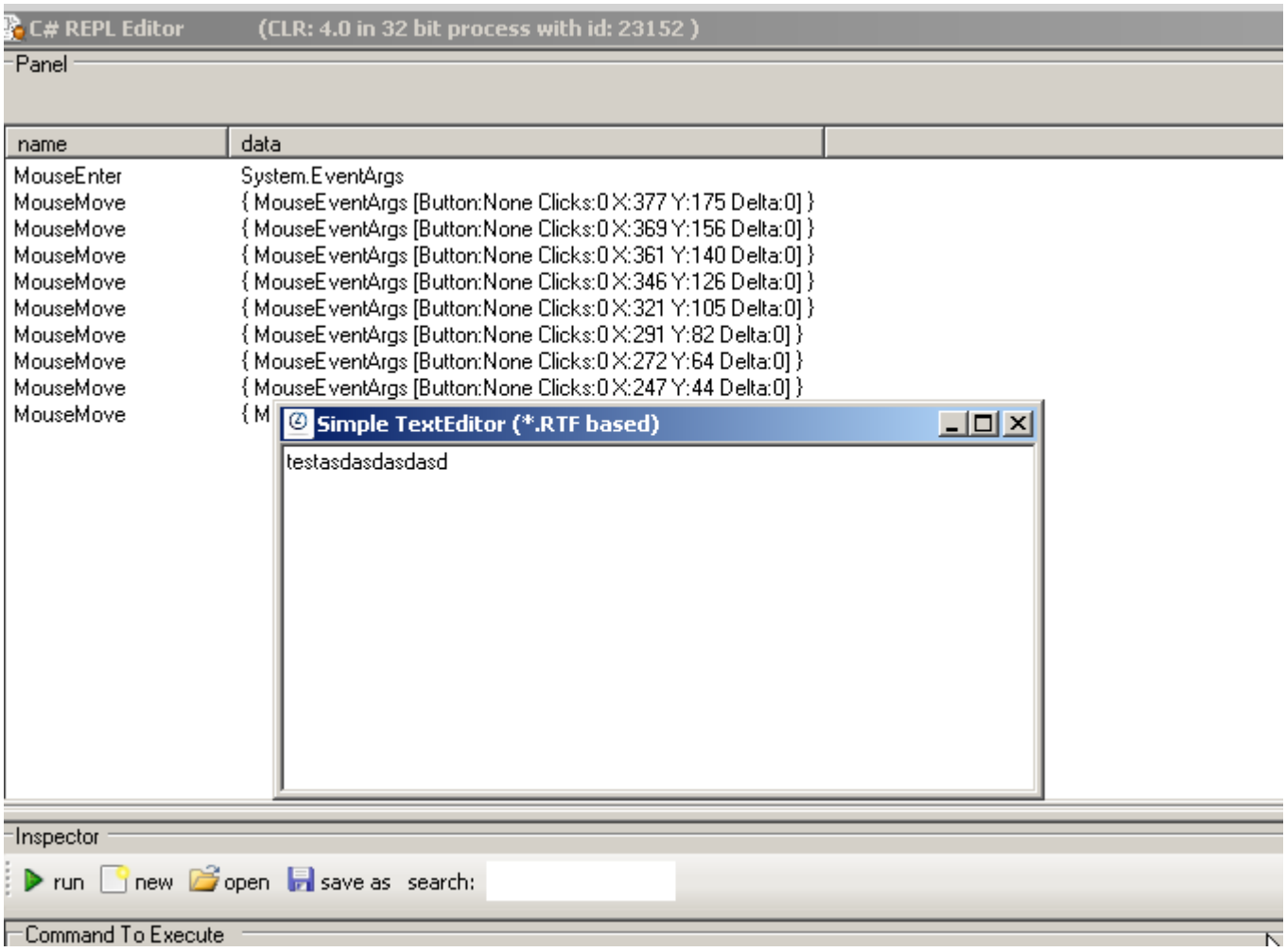
```
var simpleEditorExe = @"C:/_WorkDir/O2/O2 Install/_TempDir_v4.5.1.0/11_17_2012/Util - Simple Text Editor
[18704]\Util - Simple Text Editor.exe";

var targetProcess =   simpleEditorExe.startProcess();
while(targetProcess.MainWindowHandle == IntPtr.Zero) { targetProcess.sleep(250); }
var handle = targetProcess.MainWindowHandle;

ControlProxy proxy =null;
 Control eventWindow = null;

var sync = new System.Threading.AutoResetEvent(false);
var thread = O2Thread.mtaThread(
       ()=>{
                     //first time it is executed the eventWindow will be created
                     eventWindow = (Control)"ManagedSpyLib.dll".assembly().type("Desktop")
                                                 .fieldValue("eventWindow");
                     if(eventWindow.Handle.window_ThreadId() ==0)
                           "eventWindow Thread ID was 0".error();
                     proxy = ControlProxy.FromHandle(handle);
                     sync.Set();
                     Application.Run();

              });

sync.WaitOne();

eventWindow.invokeOnThread(
       ()=>{
                     var currentProxy =  proxy.Children[0];

                     currentProxy.EventFired += new ControlProxyEventHandler(
                                   (sender, args)=>{

                                                                                 var messageDetails =
"{0} : {1}".format(args.eventDescriptor.Name, args.eventArgs.ToString());
                                                                                 "message received:
{0}".debug(messageDetails);
                                                                          });


                     currentProxy.SetValue("Text", "test");
                     foreach (EventDescriptor @event in currentProxy.GetEvents())
                           currentProxy.SubscribeEvent(@event.Name);

                     System.Diagnostics.Debug.WriteLine("Configured ControlProxy");
                     "Events hook setup".info();
              });

targetProcess.WaitForExit();
thread.Abort();
return "done";

//using System.ComponentModel
//using Microsoft.ManagedSpy

//generateDebugSymbols
//O2Ref:ManagedSpyLib.dll
//O2File:API_WinAPI.cs
//O2File:Win32_Helper_Methods.cs

 //_O2Ref:C:\_WorkDir\O2\O2 Install\_TempDir_v4.5.1.0\_ToolsOrApis\ManagedSPY\ManagedSpy\Debug
\ManagedSpyLib.dll
```

Showing data in Table_List

```
var topPanel = panel.add_Panel(true);
var tableList = topPanel.add_TableList().add_Columns("name", "data");

{...}
eventWindow.invokeOnThread(
       ()=>{
                     var currentProxy =  proxy.Children[0];

                     currentProxy.EventFired += new ControlProxyEventHandler(
                                   (sender, args)=>{
                                                                          tableList.add_Row
(args.eventDescriptor.Name, args.eventArgs.ToString());
                                                                          //     var messageDetails =
"{0} : {1}".format(args.eventDescriptor.Name, args.eventArgs.ToString());
                                                                          //     "message received:
```

```
{0}".debug(messageDetails);
                                                                    });
```

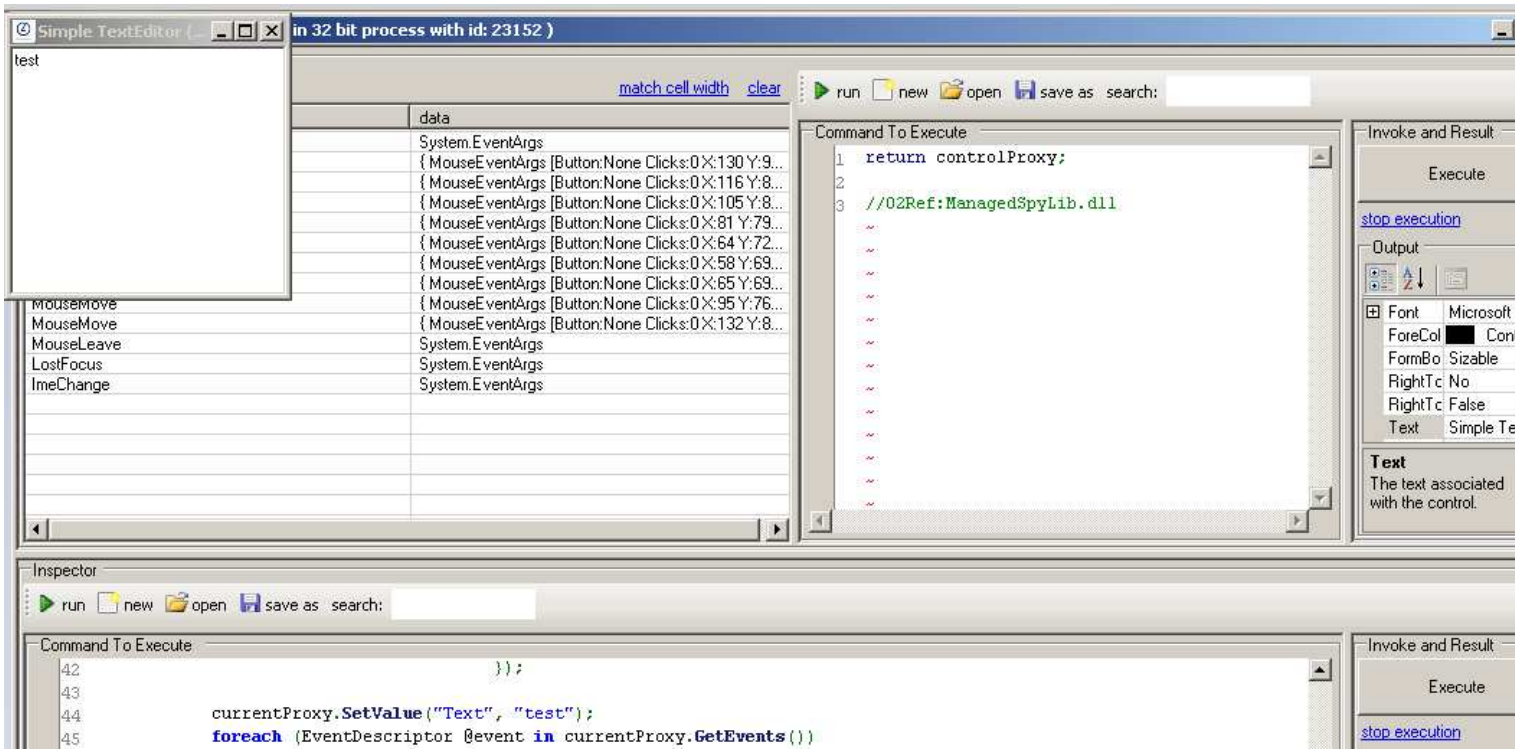
```
{...}
```



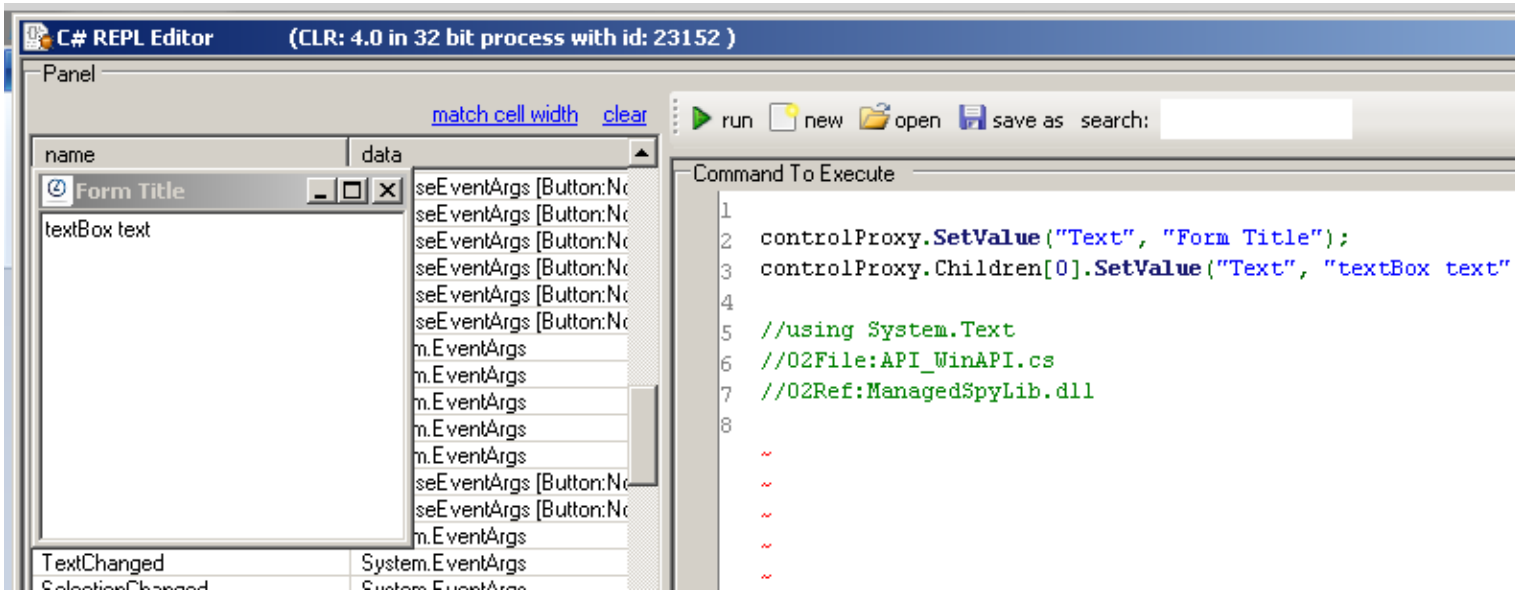
Adding a Script_Me C# REPL and showing the new process window always on top

```
var topPanel = panel.add_Panel(true);
var tableList = topPanel.add_TableList().add_Columns("name", "data");

var simpleEditorExe = @"C:/_WorkDir/O2/O2 Install/_TempDir_v4.5.1.0/11_17_2012/Util - Simple Text Editor
[18704]\Util - Simple Text Editor.exe";

var targetProcess =   simpleEditorExe.startProcess();

while(targetProcess.MainWindowHandle == IntPtr.Zero) { targetProcess.sleep(250); }
var handle = targetProcess.MainWindowHandle;

handle.window_AlwaysOnTop()
        .window_Move(000,2,200,200);

ControlProxy proxy =null;
 Control eventWindow = null;

var sync = new System.Threading.AutoResetEvent(false);
var thread = O2Thread.mtaThread(
      ()=>{
                  //first time it is executed the eventWindow will be created
                  eventWindow = (Control)"ManagedSpyLib.dll".assembly().type("Desktop")
                                                      .fieldValue("eventWindow");
                  if(eventWindow.Handle.window_ThreadId() ==0)
                        "eventWindow Thread ID was 0".error();
                  proxy = ControlProxy.FromHandle(handle);
                  sync.Set();
                  Application.Run();
```

```
            });

sync.WaitOne();
var scriptEditor = tableList.insert_Right_Script_Me(proxy);
eventWindow.invokeOnThread(
       ()=>{
                    var currentProxy =  proxy.Children[0];

                    currentProxy.EventFired += new ControlProxyEventHandler(
                                  (sender, args)=>{
                                                                  tableList.add_Row
(args.eventDescriptor.Name, args.eventArgs.ToString());
                                                                  //     var messageDetails =
"{0} : {1}".format(args.eventDescriptor.Name, args.eventArgs.ToString());
                                                                  //     "message received:
{0}".debug(messageDetails);
                                                           });


                    currentProxy.SetValue("Text", "test");
                    foreach (EventDescriptor @event in currentProxy.GetEvents())
                          currentProxy.SubscribeEvent(@event.Name);

                    System.Diagnostics.Debug.WriteLine("Configured ControlProxy");
                    "Events hook setup".info();
            });

targetProcess.WaitForExit();
thread.Abort();
return "done";

//using System.ComponentModel
//using Microsoft.ManagedSpy

//generateDebugSymbols
//O2Ref:ManagedSpyLib.dll
//O2File:API_WinAPI.cs
//O2File:API_WinAPI_ExtensionMethods.cs

 //_O2Ref:C:\_WorkDir\O2\O2 Install\_TempDir_v4.5.1.0\_ToolsOrApis\ManagedSPY\ManagedSpy\Debug
\ManagedSpyLib.dll
```



Changing Form and TextBox value from the Script_Me



Now we can script the subscription and unsubscription of events

```
tableList.clearRows();
var currentProxy = controlProxy.Children[0];

foreach (EventDescriptor @event in currentProxy.GetEvents())
      currentProxy.UnsubscribeEvent(@event.Name);

currentProxy.SubscribeEvent("TextChanged");
currentProxy.SubscribeEvent("MouseClick");
currentProxy.SetValue("Text", "123456");
return currentProxy;
//O2Ref:ManagedSpyLib.dll
//using System.ComponentModel
```