ManagedSpy - Using O2 VS Extension to create ManagedSpy UserControls or Forms inside VisualStudio

Installing O2 FluentSharp C# REPL VisualStudio extension : http://visualstudiogallery.msdn.microsoft.com/295fa0f6-37d1-49a3-b51d-ea4741905dc2

Extensions > Tools > VisualStudio C# REPL - O2 Platform

VisualStudio C# REPL - O2 Platform Free

VisualStudio 2010 Extension for the which provides a real-time C# REPL for VisualStudio (based on the OWASP O2 Platform and FluentSharp API) 4.4.14 - update to latest version of FluentSharp dlls

CREATED BY	Dinis Cruz (OWASP O2 Platform)	LAST UPDATED	11/6/2012
REVIEWS	★★★★★ (0) Review	VERSION	4.4.14
SUPPORTS	Visual Studio 2010	LICENSE	View
DOWNLOADS	Download (309)	SHARE	
		FAVORITES	Add To Favorites
TAGS	Security, OWASP, REPL		

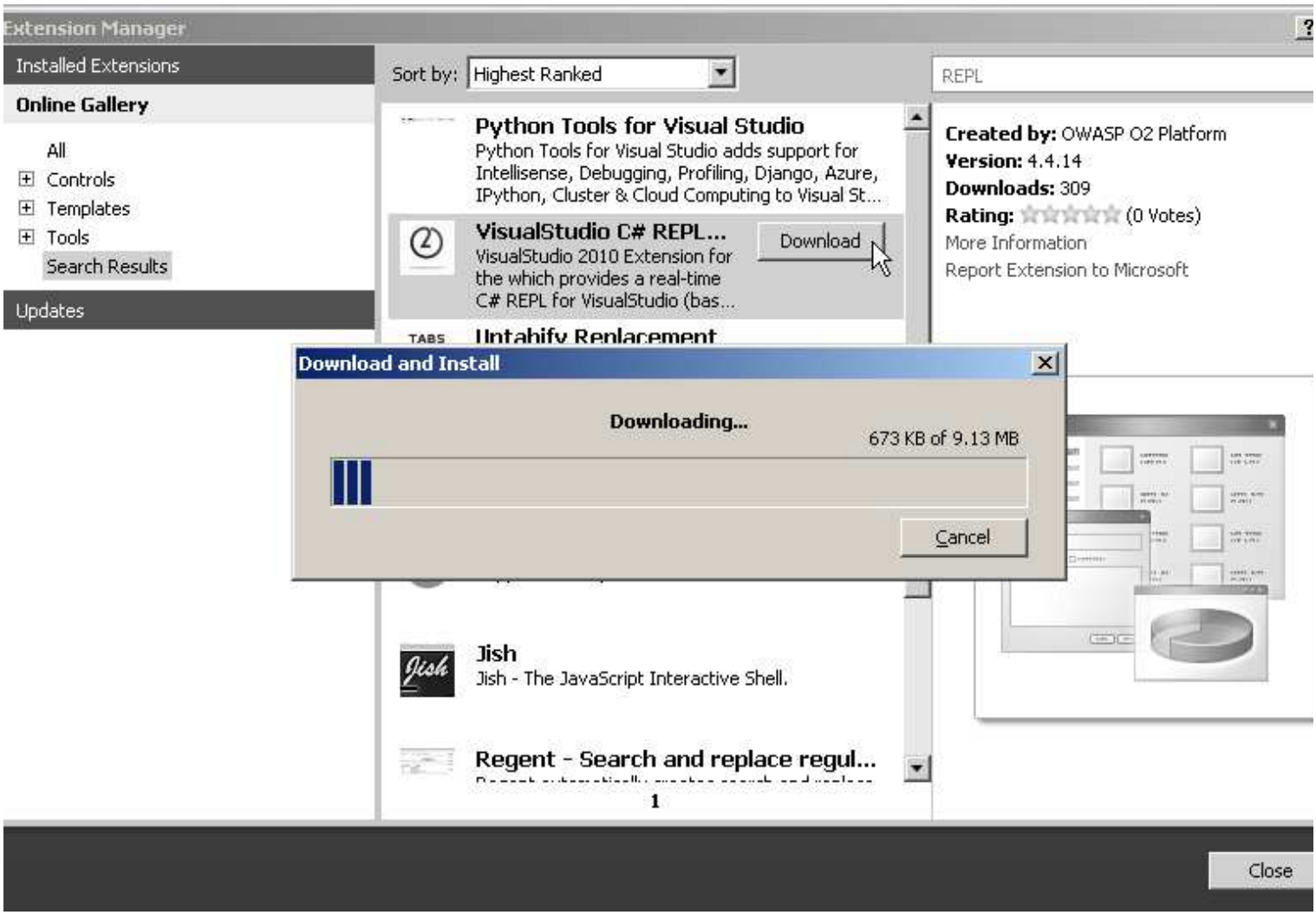
DESCRIPTION REVIEWS Q AND A

This extension provides a C# REPL Scripting environment (based on O2 Platforms's FluentSharp APIs).

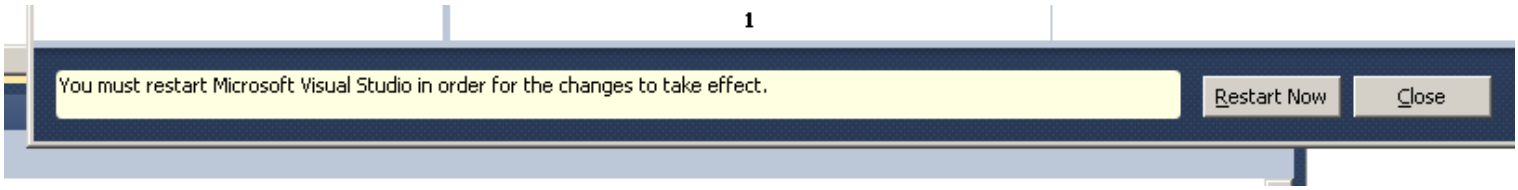
In addition to being able to write and execute quick C# snippets (in a REPL environment), **you can program VisualStudio IDE in real time!**

Here is a code sample that shows how to use FluentSharp's VisualStudio API to manipulate multiple parts of the VisualStudio IDE:

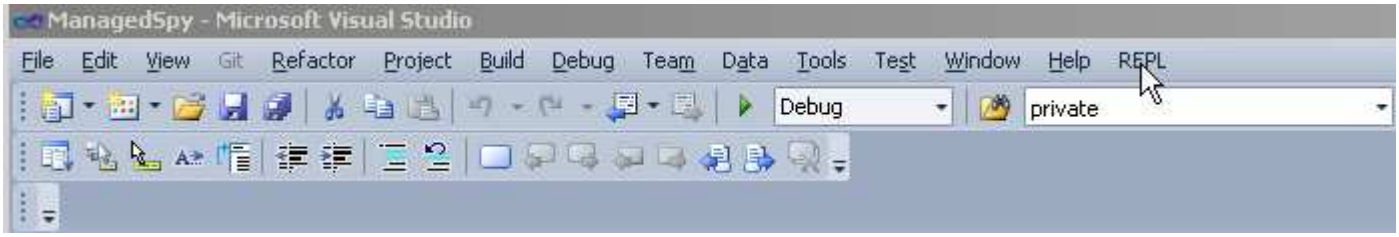
Which can be installed from the VisualStudio Extension Manager



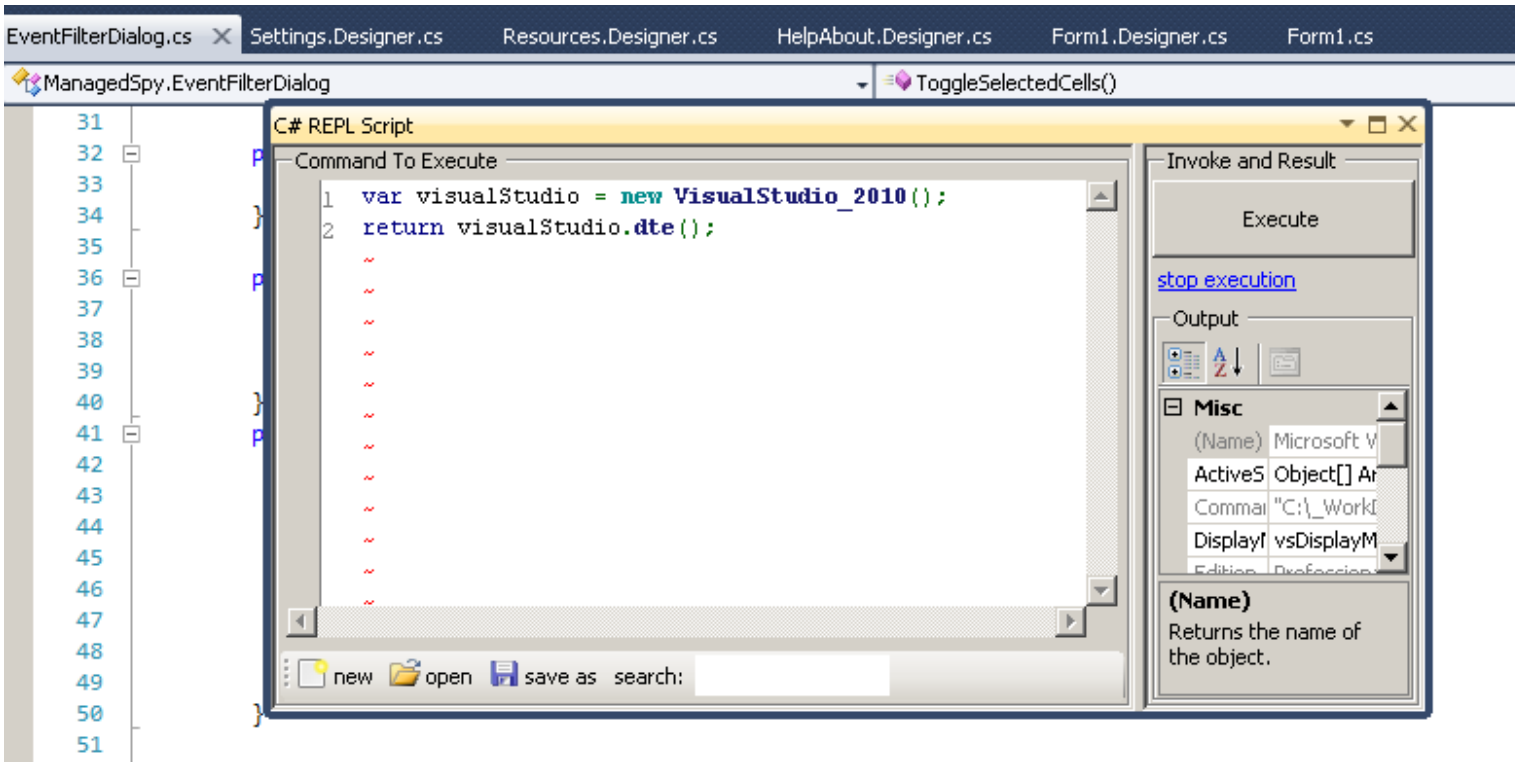
After install, restart visual studio:



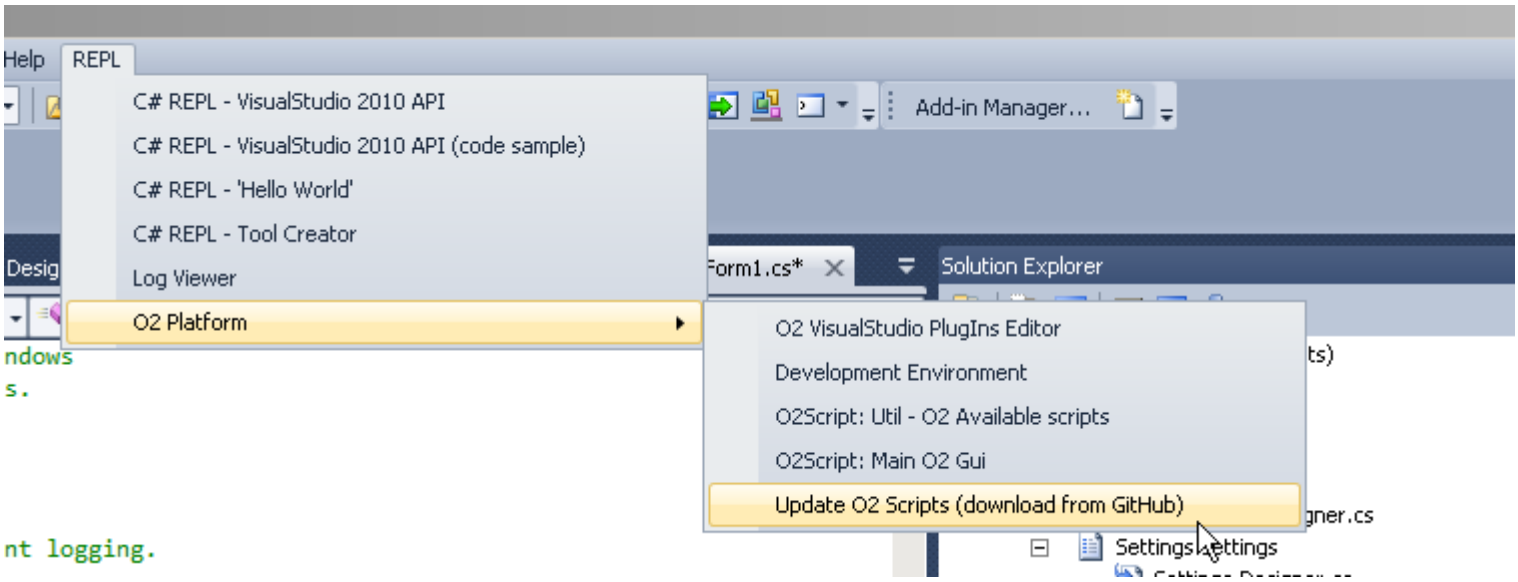
After restart you should have a new REPL menu:

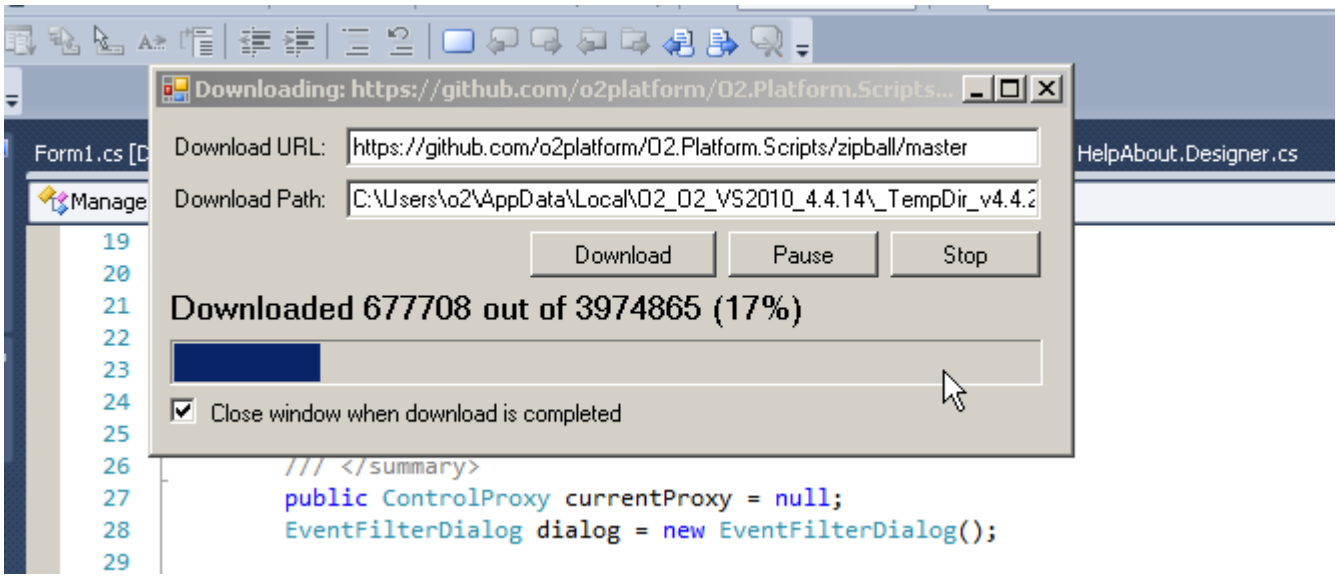


Which will give you a the C# REPL environment for VisualStudio:

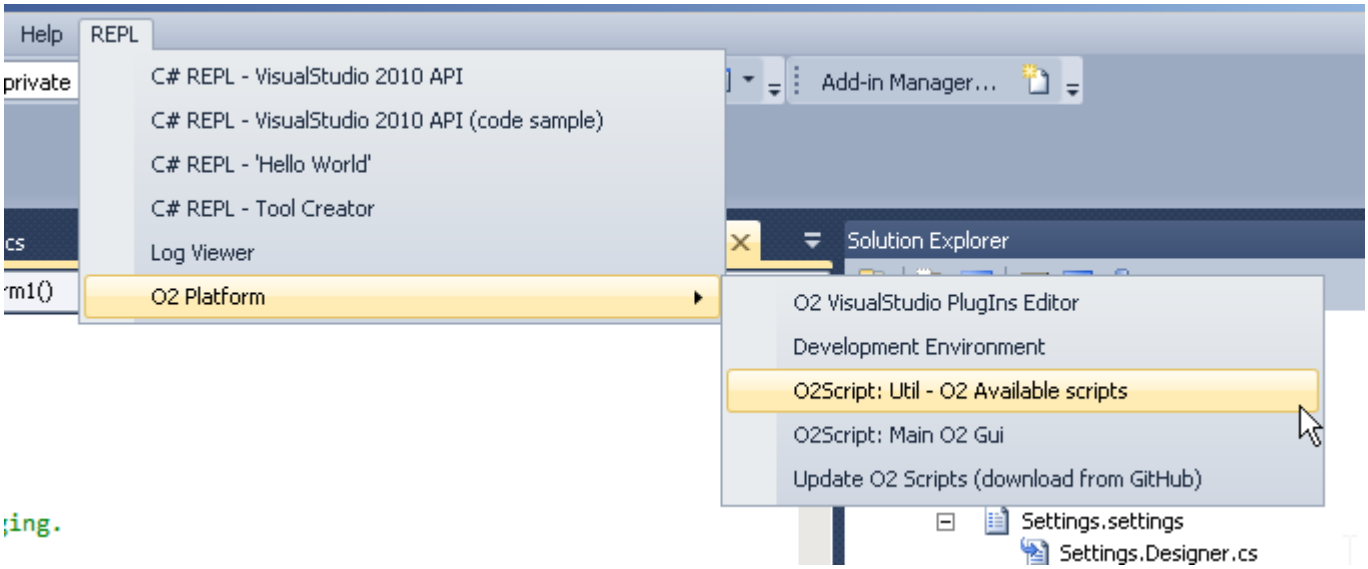


This version of the VS Extension doesn't include the O2.Platform.Scripts (from https://github.com/o2platform/O2.Platform.Scripts) **so let's download them:**

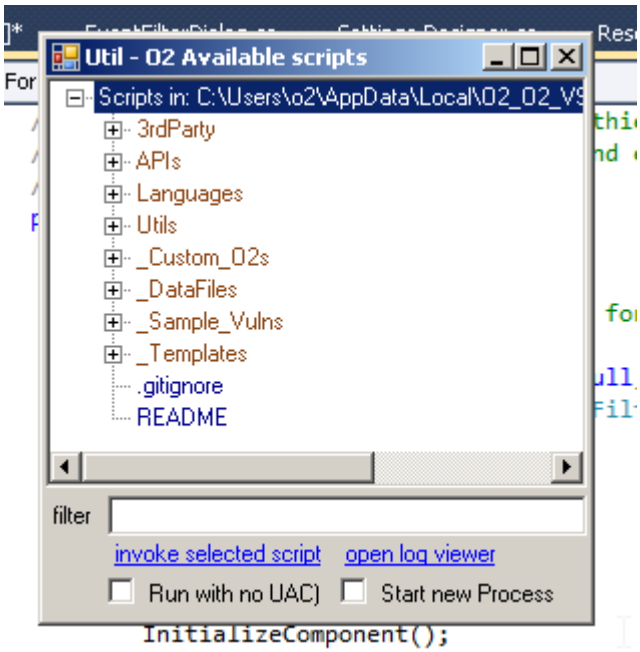




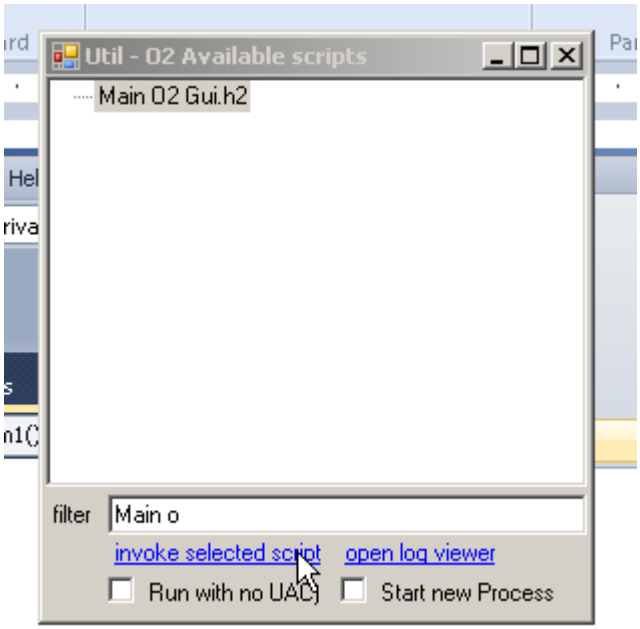
Once the download is complete you can view all scripts via:



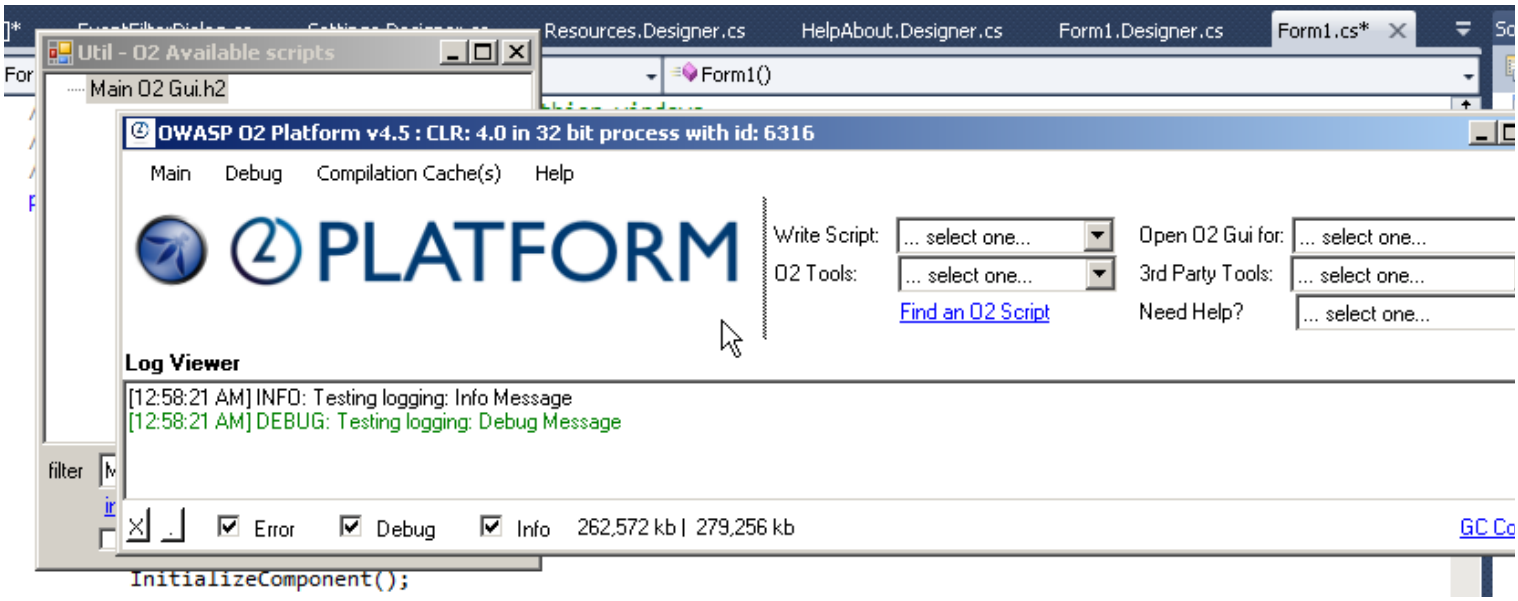
which looks like this



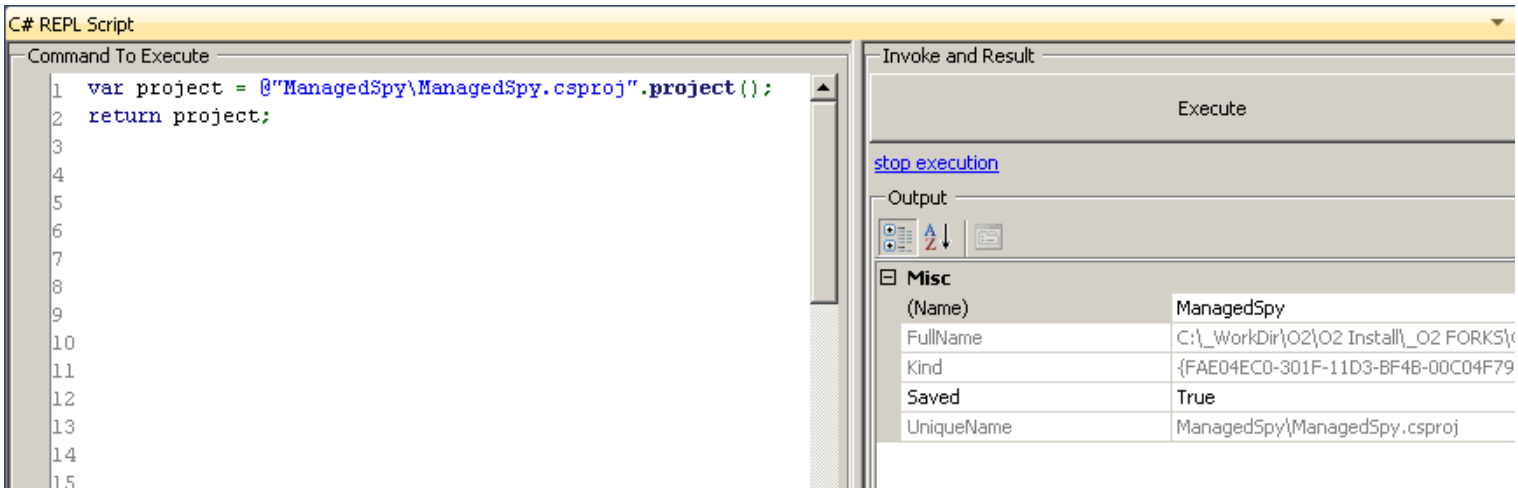
Note that the main O2 Gui is itself a script :)



double click on script name or on the *invoke selected script* link to execute it



The VisualStudio C# REPL, give us access to DTE object for ManagedSpy project



And the Path to the compiled assembly

C# REPL Script

Command To Execute

```
Project project = @"ManagedSpy\ManagedSpy.csproj".project()
return project.pathTo_CompiledAssembly();

//using EnvDTE;
```

Invoke and Result

Execute

stop execution

Output

```
C:\_WorkDir\O2\O2 Install\_O2 FORKS\O2_Fork_ManagedSpy\ManagedSpy
\bin\Debug\ManagedSpy.exe
```

which for reference was calculated like this:

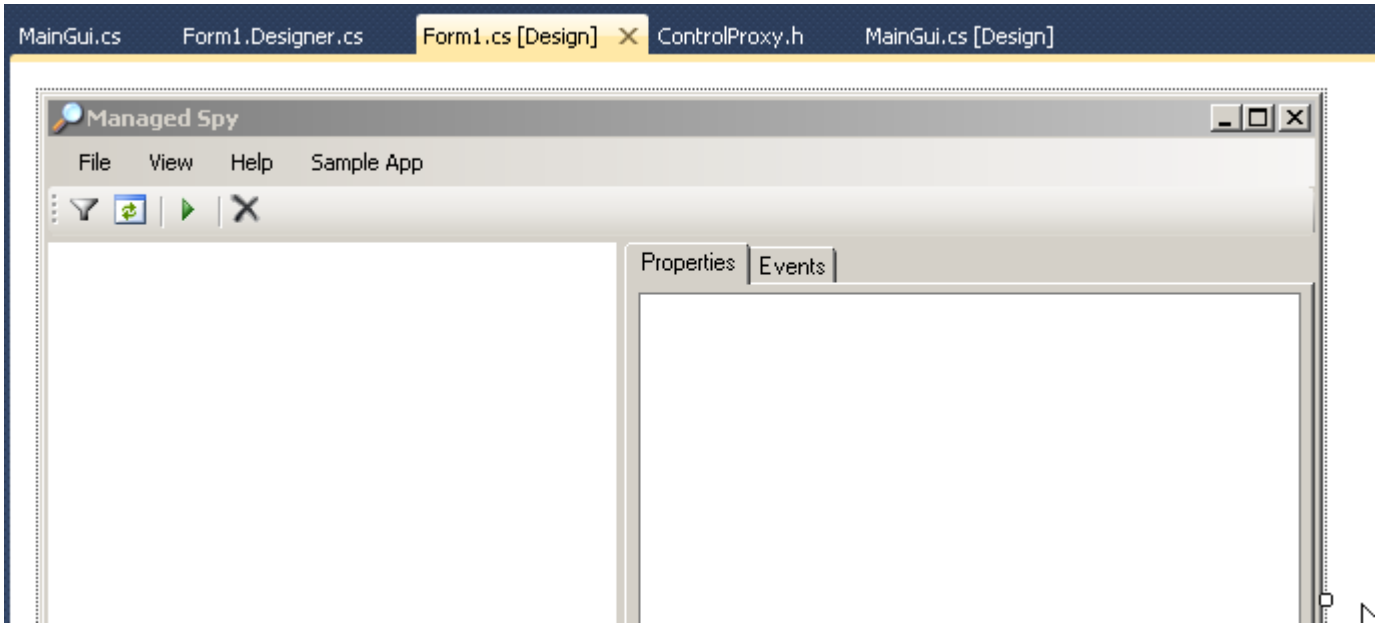
https://github.com/o2platform/O2.FluentSharp/blob/master/O2.FluentSharp.VisualStudio_2010/ExtensionMethods/VisualStudio_2010_ExtensionMethods.cs

markl...

```
        public static string pathTo_CompiledAssembly(this Project project)
        {
                try
                {
                        var relativeOutputPath = project.ConfigurationManager.ActiveConfiguration.Properties.Item("O
                                                                                .Value.str();
                        var projectPath =       project.ProjectItems.ContainingProject.FullName.parentFolder();
                        var outputPath  = projectPath.pathCombine(relativeOutputPath);
                        var outputFileName = project.Properties.Item("OutputFileName").Value.str();
                        var fullPathToCompiledAssembly = outputPath.pathCombine(outputFileName);
                        return fullPathToCompiledAssembly;
                }
                catch(Exception ex)
                {
                        ex.log("[VisualStudio_2010][pathTo_CompiledAssembly]");
                        return null;
                }
        }
```

Moving back into the main VS enviroment:

if you look the Form Designer for *Form1* (which uses the *MainGui.cs* UserControl), notice that the *SimpleApp* Menu Item was added)



But since that code uses the O2 FluentSharp APIs

```
MainGui.cs*   Form1.Designer.cs   Form1.cs [Design]   ControlProxy.h   MainGui.cs [Design]*
ManagedSpy.MainGui                                   MainGui()
1   using ...
13
14  namespace ManagedSpy
15  {
16      /// <summary> ...
23      public partial class MainGui : UserControl
24      {
25          public ControlProxy currentProxy = null;
26          EventFilterDialog dialog = new EventFilterDialog();
27
28          public MainGui()
29          {
30              InitializeComponent();
31              //adding O2
32              this.add_ExtraMenuItems();
33
34          }
```

That means that the code in the MainGui's contructor is invoked (on Build or Form1 rendering)

which is something that we can confirm like this (note the info message on the O2 Log Viewer that is running inside VisualStudio)

```
MainGui.cs   Form1.Designer.cs   Form1.cs [Design]
ManagedSpy.MainGui               MainGui()
1   using ...
13
14  namespace ManagedSpy
15  {
16      /// <summary> ...
23      public partial class MainGui : UserControl
24      {
25          public ControlProxy currentProxy = null;
26          EventFilterDialog dialog = new EventFilter
27
28          public MainGui()
29          {
30              InitializeComponent();
31              //adding O2
32              "This code was invoked".info();
33              this.add_ExtraMenuItems();
34
35          }
```

```
Solution Explorer
Solution 'ManagedSpy' (2 projects)
  ManagedSpy
    Properties
    References
    O2_Platform
      MainGui_ExtensionMethods.cs
    UserControls
      ClickToolStrip.cs
      MainGui.cs
        MainGui.Designer.cs
Solution Explorer   Team Explorer   Class View
O2 LogViewer
Log Viewer
[2:18:55 AM] INFO: This code was invoked
```

we can define a global (to O2's FluentSharp objects) variable to hold an *Action* object

```
C# REPL Script
Command To Execute
Action callbackFromVS =
    ()=>{
            "I was called on MainGui ctor".info();
        };

"onMainGuiCtor".o2Cache(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeName();
~
~
~
~
```

```
Invoke and Result
Execute
stop execution
Output
Action
```

which is available on the MainGui ctor

```
28        public MainGui()
29        {
30            InitializeComponent();
31            //adding O2
32            this.add_ExtraMenuItems();
33            "onMainGuiCtor".o2Cache().typeName().info();
34        }
35
100 %
```

```
O2 LogViewer
Log Viewer
[2:26:49 AM] INFO: Action
```

Which can be invoked like this:

```
28        public MainGui()
29        {
30            InitializeComponent();
31            //adding O2
32            this.add_ExtraMenuItems();
33            var callbackFromVS = (Action)"onMainGuiCtor".o2Cache();
34            callbackFromVS();
35        }
100 %
```

```
O2 LogViewer
Log Viewer
[2:27:51 AM] INFO: I was called on MainGui ctor
[2:26:49 AM] INFO: Action
```

Where it gets more interresting is if we pass an object to the callback:

C# REPL Script

Command To Execute

```
Action<object> callbackFromVS =
    (_object)=>{
                "I was called on MainGui ctor with: {0}".info(_object);
                };
"onMainGuiCtor".o2Cache(null); // clear cache so that object is created bellow
"onMainGuiCtor".o2Cache<Action<object>>(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeFullName();
~
~
~
```

Invoke and Result

Execute

stop execution

Output

```
System.Action`1
[[System.Object,
mscorlib, Version=
4.0.0.0,
Culture=neutral,
PublicKeyToken=b77a
561934e089]]
```

like the MainGui object created by VisualStudio

```
28        public MainGui()
29        {
30            InitializeComponent();
31            //adding O2
32            this.add_ExtraMenuItems();
33            var callbackFromVs = (Action<object>)"onMainGuiCtor".o2Cache();
34            callbackFromVs(this);
35        }
```

100 %

O2 LogViewer

Log Viewer

```
[2:32:47 AM] INFO: I was called on MainGui ctor with: ManagedSpy.MainGui
```

Lets find out where this object's assembly actually is located:

```
Action<object> callbackFromVS =
       (_object)=>{
                     "I was called on MainGui ctor with: {0}".info(_object);
                     "Assembly location: {0}".info(_object.type().assemblyLocation());
                   };

"onMainGuiCtor".o2Cache(null); // clear cache so that object is created bellow
"onMainGuiCtor".o2Cache<Action<object>>(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeFullName();
```

which when build points to

```
28        public MainGui()
29        {
30            InitializeComponent();
31            //adding O2
32            this.add_ExtraMenuItems();
33            var callbackFromVs = (Action<object>)"onMainGuiCtor".o2Cache();
34            callbackFromVs(this);
35        }
```

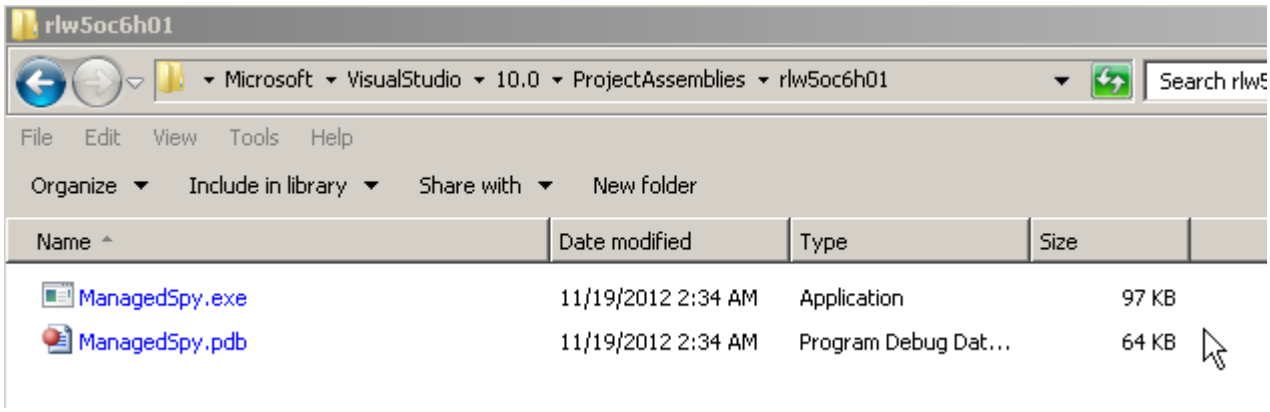
100 %

O2 LogViewer

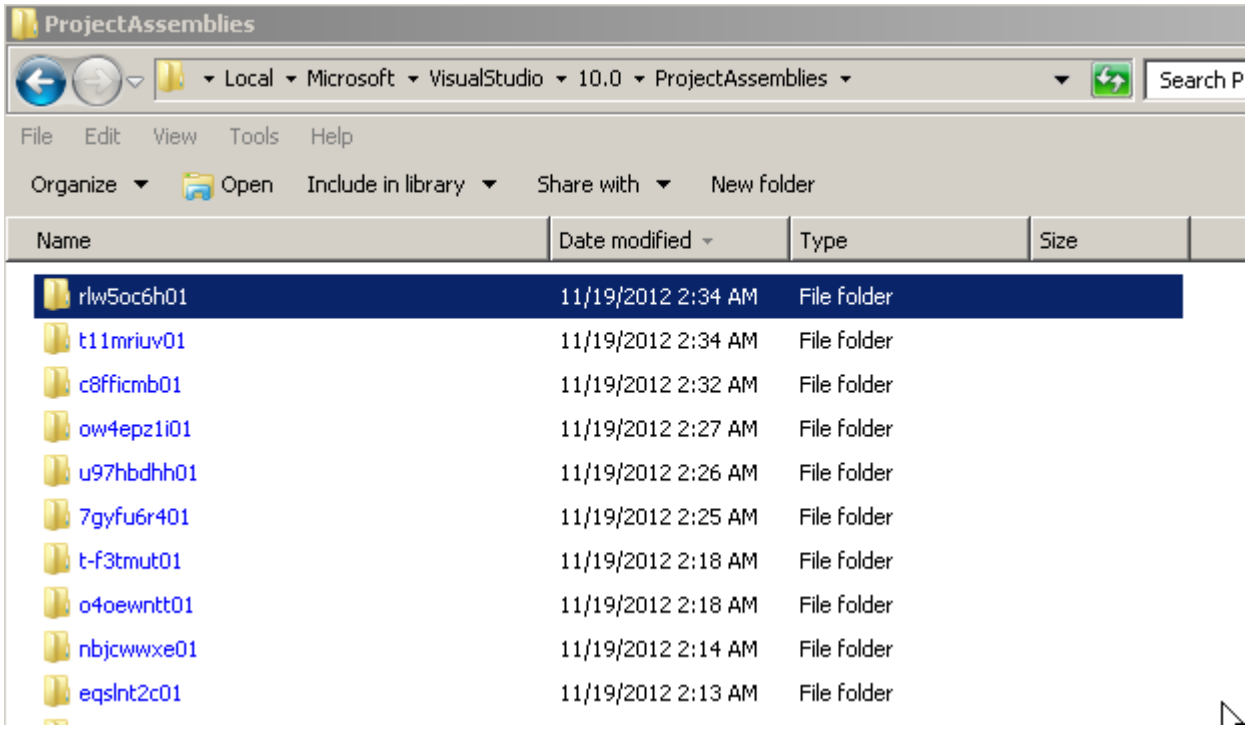
Log Viewer

```
[2:34:15 AM] INFO: Assembly location: C:\Users\o2\AppData\Local\Microsoft\VisualStudio\10.0\ProjectAssemblies\rlw5oc6h01\ManagedSpy.exe
[2:34:15 AM] INFO: I was called on MainGui ctor with: ManagedSpy.MainGui
[2:34:02 AM] INFO: Assembly location: C:\Users\o2\AppData\Local\Microsoft\VisualStudio\10.0\ProjectAssemblies\t11mriuv01\ManagedSpy.exe
[2:34:02 AM] INFO: I was called on MainGui ctor with: ManagedSpy.MainGui
```

i.e. this folder:

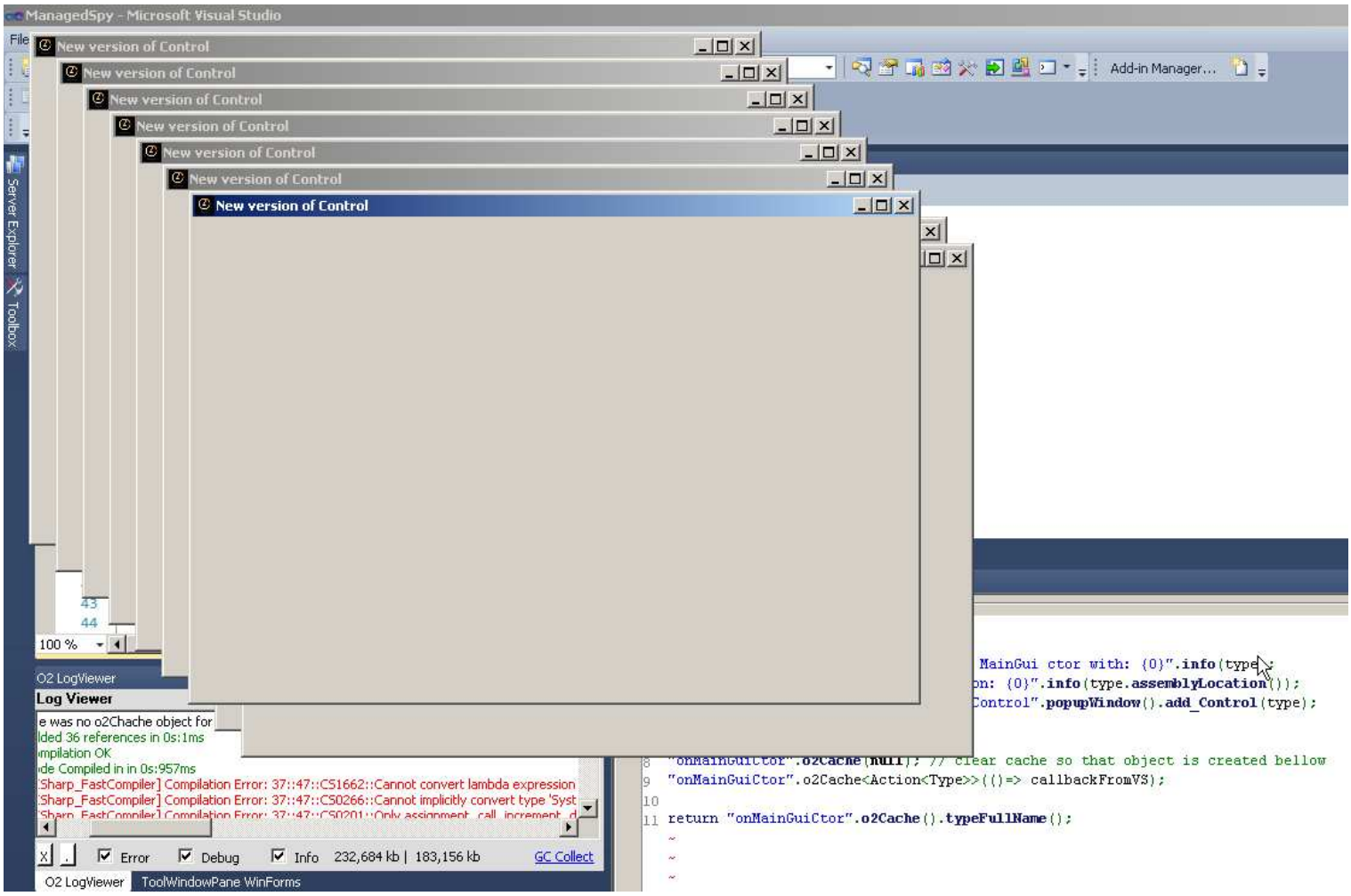


which looks like contains the latest version of the controls created:

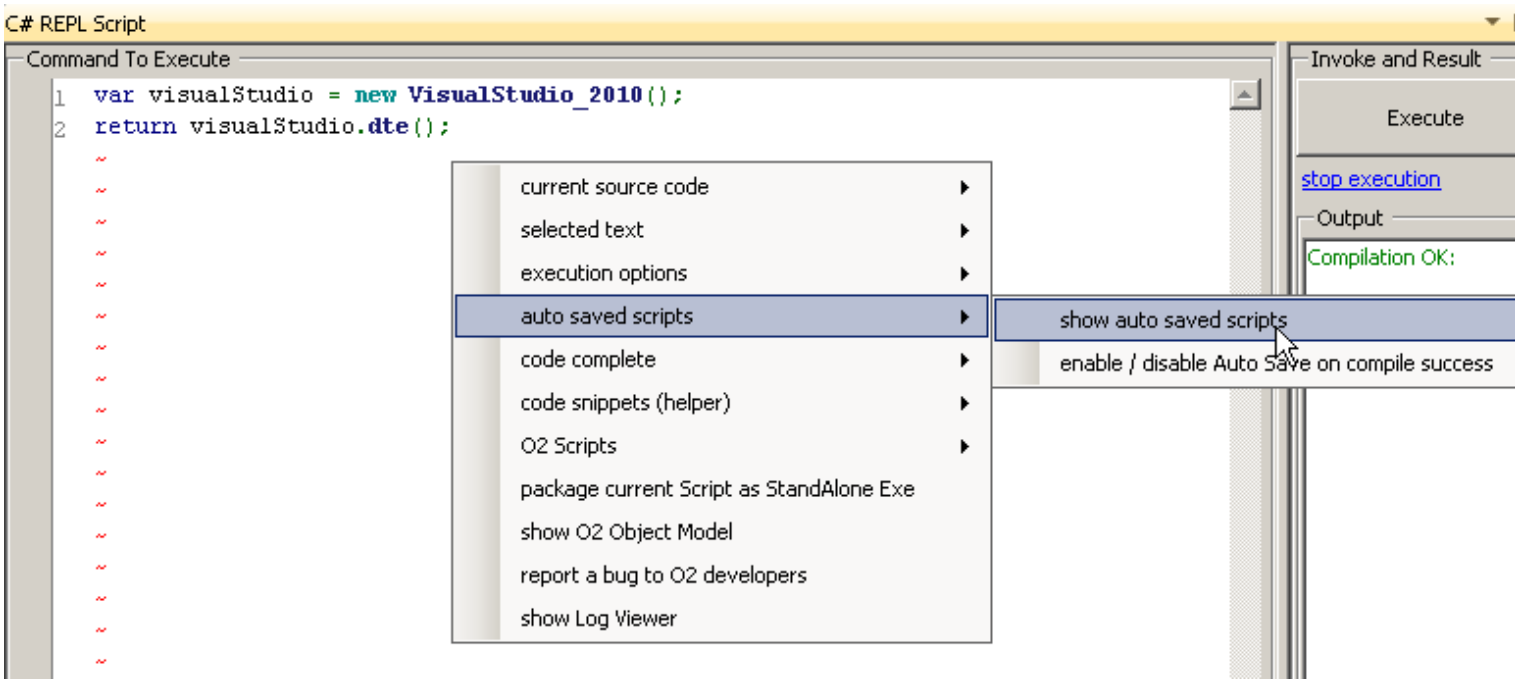


Note if you crash VisualStudio (like with this code that created a recursive creation of windows)

```
Action<Type> callbackFromVS =
       (type)=>{
                                    "I was called on MainGui ctor with: {0}".info(type);
                                    "Assembly location: {0}".info(type.assemblyLocation());

                                    "New version of Control".popupWindow().add_Control(type);
                        };

"onMainGuiCtor".o2Cache(null); // clear cache so that object is created bellow
"onMainGuiCtor".o2Cache<Action<Type>>(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeFullName();
```



(on restart) use the *auto saved scripts*



to recover your script :)

```
C# REPL Script
Command To Execute
var visualStudio = new VisualStudio_2010();
return visualStudio.dte();
Invoke and Result
Execute
stop execution
Output
(2 projects)
Util - Search AutoSaved Scripts (starting with Today)
Files
(02h 31s 15ms , 19-11-12).cs
(02h 31s 16ms , 19-11-12).cs
(02h 31s 21ms , 19-11-12).cs
(02h 33s 32ms , 19-11-12).cs
(02h 33s 33ms , 19-11-12).cs
(02h 33s 53ms , 19-11-12).cs
(02h 37s 16ms , 19-11-12).cs
(02h 37s 48ms , 19-11-12).cs
(02h 38s 13ms , 19-11-12).cs
(02h 38s 34ms , 19-11-12).cs
(02h 38s 45ms , 19-11-12).cs
(02h 40s 06ms , 19-11-12).cs
(02h 40s 08ms , 19-11-12).cs
(02h 40s 09ms , 19-11-12).cs
(02h 40s 33ms , 19-11-12).cs
(02h 40s 36ms , 19-11-12).cs
(02h 40s 38ms , 19-11-12).cs
new
open
Saved File Contents
Action<Type> callbackFromVS =
    (type)=>{
                "I was called on MainGui ctor with: {0}".info(type);
                "Assembly location: {0}".info(type.assemblyLocation());
                "New version of Control".popupWindow().add_Control(type);
            };

"onMainGuiCtor".o2Cache(null); // clear cache so that object is created bellow
"onMainGuiCtor".o2Cache<Action<Type>>(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeFullName();
```

The recursive loop was created because we need to check if the ctor is running in the *DesignMode*:

```
Form1.Designer.cs   Form1.cs [Design]   ControlProxy.h   MainGui.cs*   MainGui.cs [Design]*
ManagedSpy.MainGui                      MainGui()
Server Explorer
Toolbox
25          public ControlProxy currentProxy = null;
26          EventFilterDialog dialog = new EventFilterDialog();
27
28          public MainGui()
29          {
30              InitializeComponent();
31              //adding O2
32              this.add_ExtraMenuItems();
33
34              if (this.DesignMode)
35              {
36                  var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cache();
37                  callbackFromVs(this.type());
38              }
39          }
```

Unfortunately that will not work since at this stage the DesignMode is also false.

```
31          public MainGui()
32          {
33              InitializeComponent();
34              //adding O2
35              this.add_ExtraMenuItems();
36
37              "{0}".info(this.DesignMode);
100 %
O2 LogViewer
Log Viewer
[3:04:05 AM] INFO: False
```

So lets take a look at the Current StackTrace

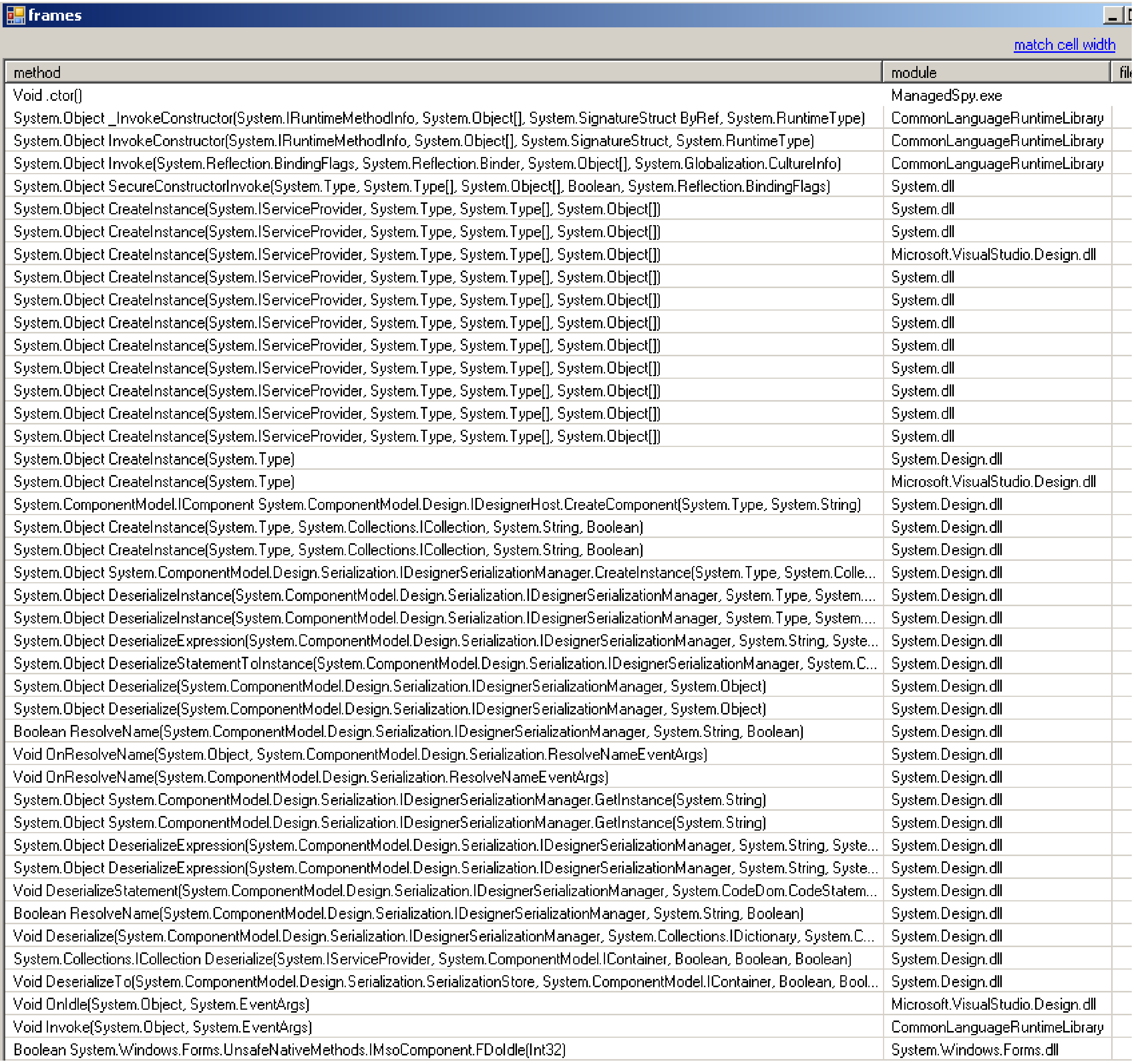
```
public MainGui()
 {
     InitializeComponent();
     //adding O2
     this.add_ExtraMenuItems();

     "{0}".info(this.DesignMode);

     "frames".popupWindow(1000,300).add_TableList()
             .show(from frame in new StackTrace().GetFrames()
                   select new { method = frame.GetMethod(),
                                module = frame.GetMethod().Module,
                                file = frame.GetFileName(),
                                line = frame.GetFileLineNumber()
                              });
```

which since placed in the MainGui.cs ctor

```
Form1.cs [Design]   ClickToolStrip.cs   Form1.Designer.cs   ControlProxy.h   MainGui.cs   MainGui.cs [D
ManagedSpy.MainGui                                          MainGui()
29          EventFilterDialog dialog = new EventFilterDialog();
30
31          public MainGui()
32          {
33              InitializeComponent();
34              //adding O2
35              this.add_ExtraMenuItems();
36
37              "{0}".info(this.DesignMode);
38
39              "frames".popupWindow(1000,300).add_TableList()
40                      .show(from frame in new StackTrace().GetFrames()
41                            select new { method = frame.GetMethod(),
42                                         module = frame.GetMethod().Module,
43                                         file = frame.GetFileName(),
44                                         line = frame.GetFileLineNumber()
45                                       });
```

Will be invoked on next Build :)

Here is the full stack trace when executed from Visual Studio

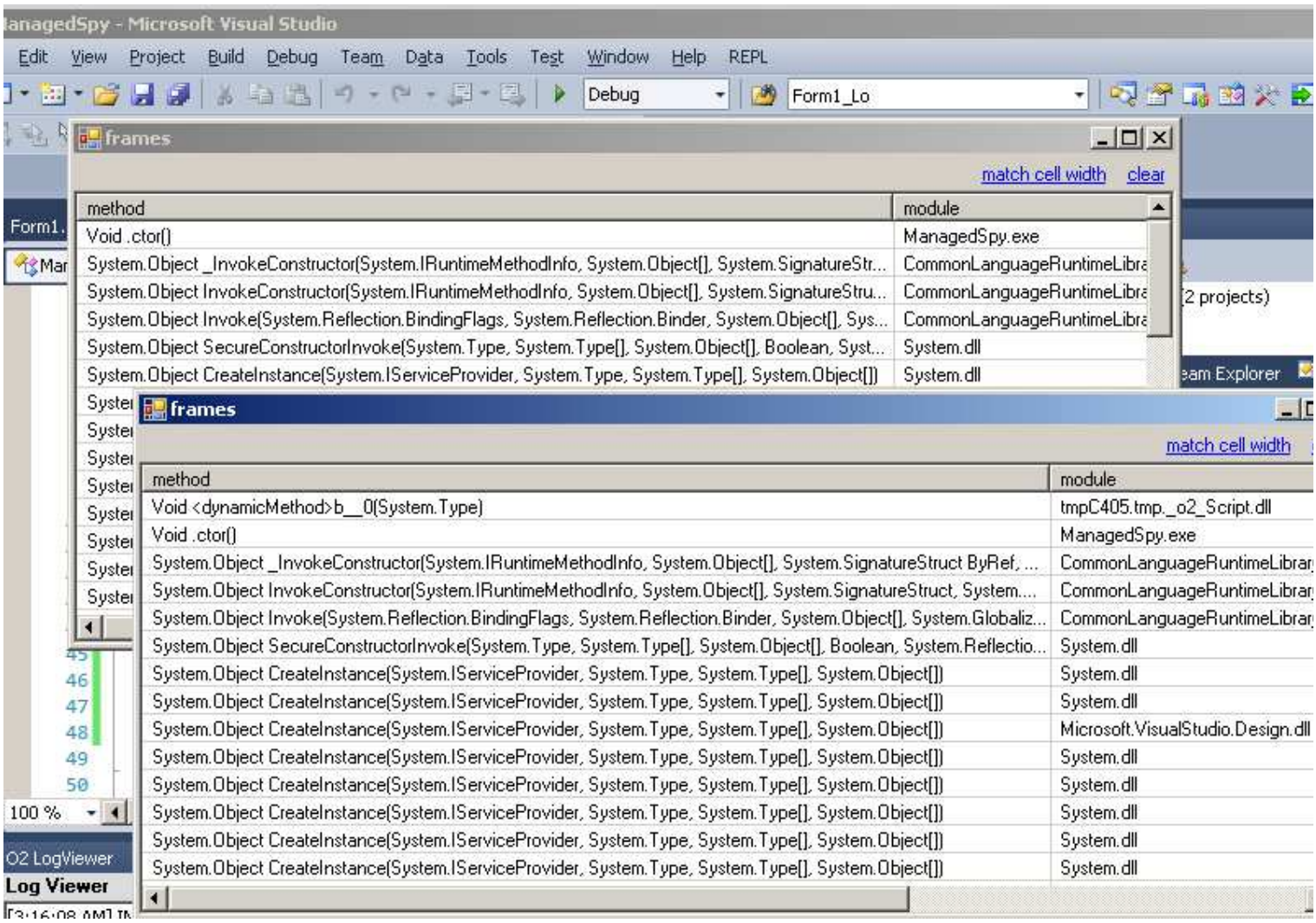


method	module	fil
Void .ctor()	ManagedSpy.exe	
System.Object _InvokeConstructor(System.IRuntimeMethodInfo, System.Object[], System.SignatureStruct ByRef, System.RuntimeType)	CommonLanguageRuntimeLibrary	
System.Object InvokeConstructor(System.IRuntimeMethodInfo, System.Object[], System.SignatureStruct, System.RuntimeType)	CommonLanguageRuntimeLibrary	
System.Object Invoke(System.Reflection.BindingFlags, System.Reflection.Binder, System.Object[], System.Globalization.CultureInfo)	CommonLanguageRuntimeLibrary	
System.Object SecureConstructorInvoke(System.Type, System.Type[], System.Object[], Boolean, System.Reflection.BindingFlags)	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	Microsoft.VisualStudio.Design.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.IServiceProvider, System.Type, System.Type[], System.Object[])	System.dll	
System.Object CreateInstance(System.Type)	System.Design.dll	
System.Object CreateInstance(System.Type)	Microsoft.VisualStudio.Design.dll	
System.ComponentModel.IComponent System.ComponentModel.Design.IDesignerHost.CreateComponent(System.Type, System.String)	System.Design.dll	
System.Object CreateInstance(System.Type, System.Collections.ICollection, System.String, Boolean)	System.Design.dll	
System.Object CreateInstance(System.Type, System.Collections.ICollection, System.String, Boolean)	System.Design.dll	
System.Object System.ComponentModel.Design.Serialization.IDesignerSerializationManager.CreateInstance(System.Type, System.Colle...	System.Design.dll	
System.Object DeserializeInstance(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.Type, System....	System.Design.dll	
System.Object DeserializeInstance(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.Type, System....	System.Design.dll	
System.Object DeserializeExpression(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.String, Syste...	System.Design.dll	
System.Object DeserializeStatementToInstance(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.C...	System.Design.dll	
System.Object Deserialize(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.Object)	System.Design.dll	
System.Object Deserialize(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.Object)	System.Design.dll	
Boolean ResolveName(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.String, Boolean)	System.Design.dll	
Void OnResolveName(System.Object, System.ComponentModel.Design.Serialization.ResolveNameEventArgs)	System.Design.dll	
Void OnResolveName(System.ComponentModel.Design.Serialization.ResolveNameEventArgs)	System.Design.dll	
System.Object System.ComponentModel.Design.Serialization.IDesignerSerializationManager.GetInstance(System.String)	System.Design.dll	
System.Object System.ComponentModel.Design.Serialization.IDesignerSerializationManager.GetInstance(System.String)	System.Design.dll	
System.Object DeserializeExpression(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.String, Syste...	System.Design.dll	
System.Object DeserializeExpression(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.String, Syste...	System.Design.dll	
Void DeserializeStatement(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.CodeDom.CodeStatem...	System.Design.dll	
Boolean ResolveName(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.String, Boolean)	System.Design.dll	
Void Deserialize(System.ComponentModel.Design.Serialization.IDesignerSerializationManager, System.Collections.IDictionary, System.C...	System.Design.dll	
System.Collections.ICollection Deserialize(System.IServiceProvider, System.ComponentModel.IContainer, Boolean, Boolean, Boolean)	System.Design.dll	
Void DeserializeTo(System.ComponentModel.Design.Serialization.SerializationStore, System.ComponentModel.IContainer, Boolean, Bool...	System.Design.dll	
Void OnIdle(System.Object, System.EventArgs)	Microsoft.VisualStudio.Design.dll	
Void Invoke(System.Object, System.EventArgs)	CommonLanguageRuntimeLibrary	
Boolean System.Windows.Forms.UnsafeNativeMethods.IMsoComponent.FDoIdle(Int32)	System.Windows.Forms.dll	

and if we add the same code to the callback (and remove the DesignMode check)

```
Action<Type> callbackFromVS =
       (type)=>{
                    "I was called on MainGui ctor with: {0}".info(type);
                    "Assembly location: {0}".info(type.assemblyLocation());
                    //"New version of Control".popupWindow().add_Control(type);

                    "frames".popupWindow(1000,300).add_TableList()
                                   .show(from frame in new StackTrace().GetFrames()
                                   select new { method = frame.GetMethod(),
                                               module = frame.GetMethod().Module,
                                               file = frame.GetFileName(),
                                               line = frame.GetFileLineNumber()
                                               });
                    };

"onMainGuiCtor".o2Cache(null); // clear cache so that object is created bellow
"onMainGuiCtor".o2Cache<Action<Type>>(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeFullName();

//using System.Diagnostics
```

we get two popup windows (that show the callback in action)



Let's create a cache referece of the type object passed to *callbackFromVS*

```
Action showFrames =
       ()=>{
                      "frames".popupWindow(1000,300).add_TableList()
                      .show(from frame in new StackTrace().GetFrames()
                            select new { method = frame.GetMethod(),
                                         module = frame.GetMethod().Module,
                                         file = frame.GetFileName(),
                                         line = frame.GetFileLineNumber()
                                       });
              };

Action<Type> callbackFromVS =
       (type)=>{
                                  "I was called on MainGui ctor with: {0}".info(type);
                                  "Assembly location: {0}".info(type.assemblyLocation());

                                  //"New version of Control".popupWindow().add_Control(type);
                                  "Putting type on newType cache object".info();
                                  "newType".o2Cache(type);
                      };

"onMainGuiCtor".o2Cache(null); // clear cache so that object is created bellow
"onMainGuiCtor".o2Cache<Action<Type>>(()=> callbackFromVS);

return "onMainGuiCtor".o2Cache().typeFullName();

//using System.Diagnostics
```

which will be triggered on next compile:

```
31          public MainGui()
32          {
33              InitializeComponent();
34              //adding O2
35              this.add_ExtraMenuItems();
36
37              "{0}".info(this.DesignMode);
38
39  /*            "frames".popupWindow(1000,300).add_TableList()
40                      .show(from frame in new StackTrace().GetFrames()
41                            select new { method = frame.GetMethod(),
42                                         module = frame.GetMethod().Module,
43                                         file = frame.GetFileName(),
44                                         line = frame.GetFileLineNumber()
45                                       });
46    */
47              var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cache();
48              callbackFromVs(this.type());
49          }
```

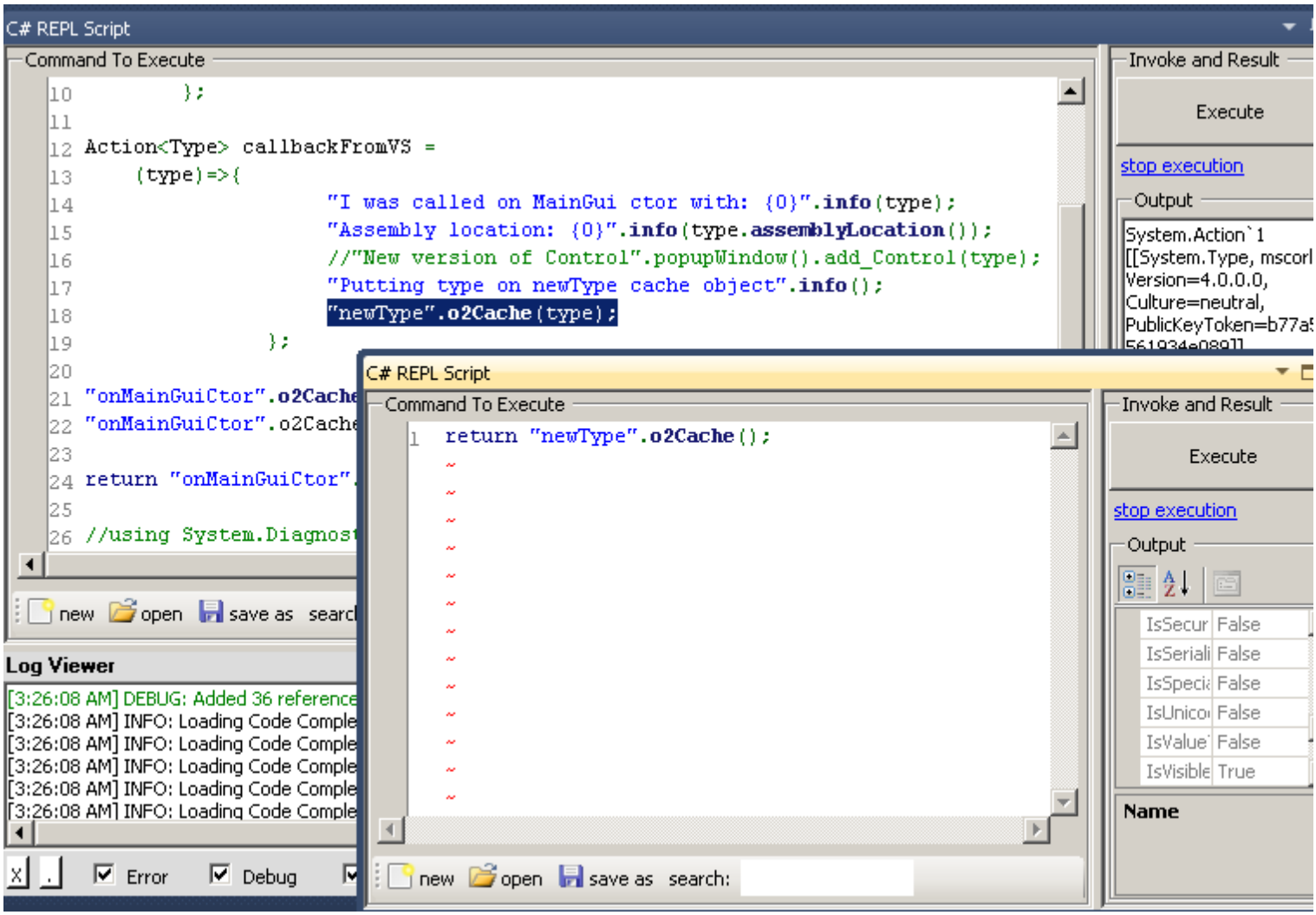
100 %

O2 LogViewer

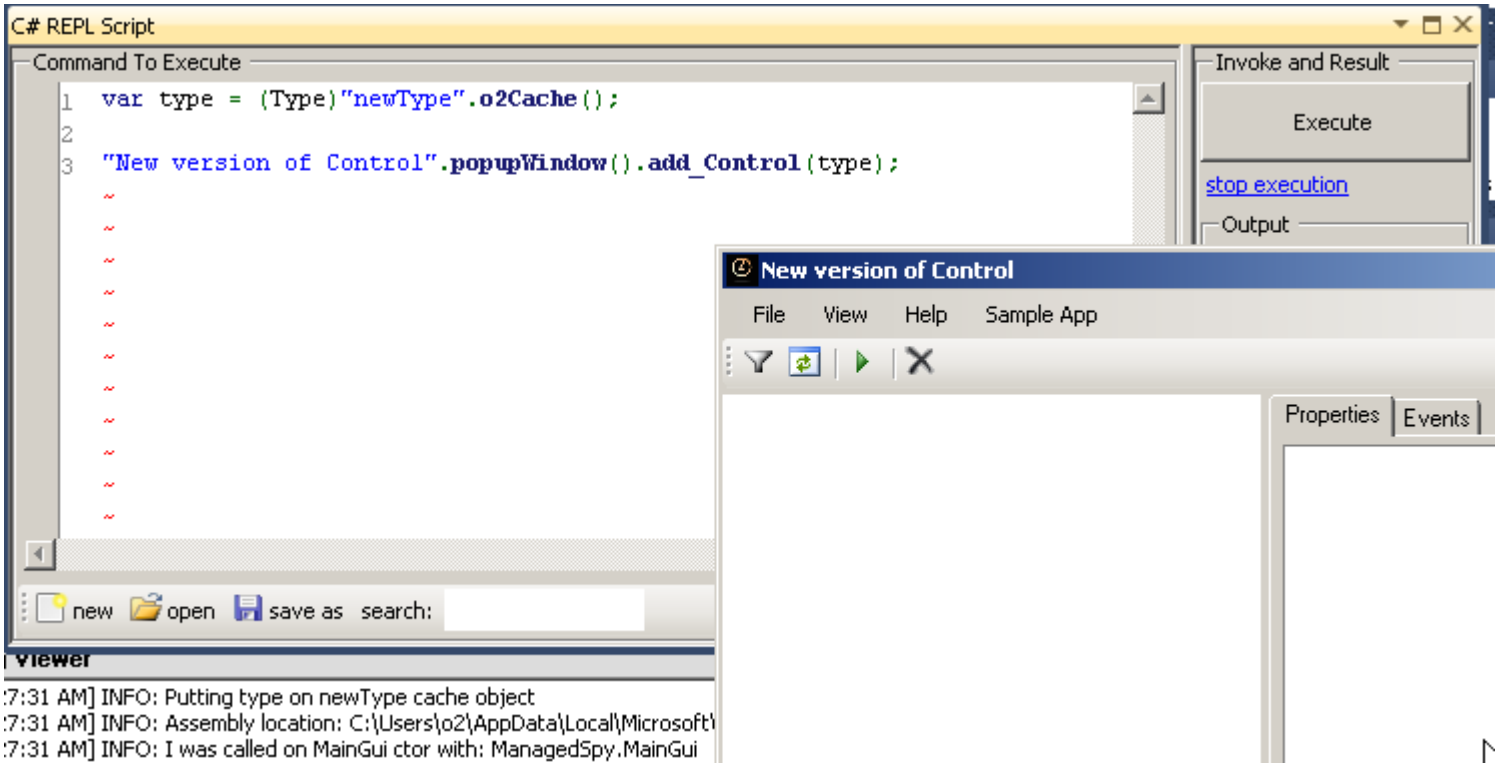
Log Viewer

```
[3:24:31 AM] INFO: Putting type on newType cache object
[3:24:31 AM] INFO: Assembly location: C:\Users\o2\AppData\Local\Microsoft\VisualStudio\10.0\ProjectAssemblies\5rx65xv
[3:24:31 AM] INFO: I was called on MainGui ctor with: ManagedSpy.MainGui
```

We can access this object from another C# REPL script environment



We can safely (with recursive loop) create this object from here (note how the callbackFromVS was still called, since the ctor of MainGui was invoked when the object was created to put on the popupWindow form)



if we add a call to showframes to callbackFromVS

```
Action<Type> callbackFromVS =
       (type)=>{
                                    "I was called on MainGui ctor with: {0}".info(type);
                                    "Assembly location: {0}".info(type.assemblyLocation());

                                    //"New version of Control".popupWindow().add_Control(type);
                                    "Putting type on newType cache object".info();
                                    "newType".o2Cache(type);
                                    showFrames();
                        };
```

and execute (again):

```
var type = (Type)"newType".o2Cache();

"New version of Control".popupWindow().add_Control(type);
```

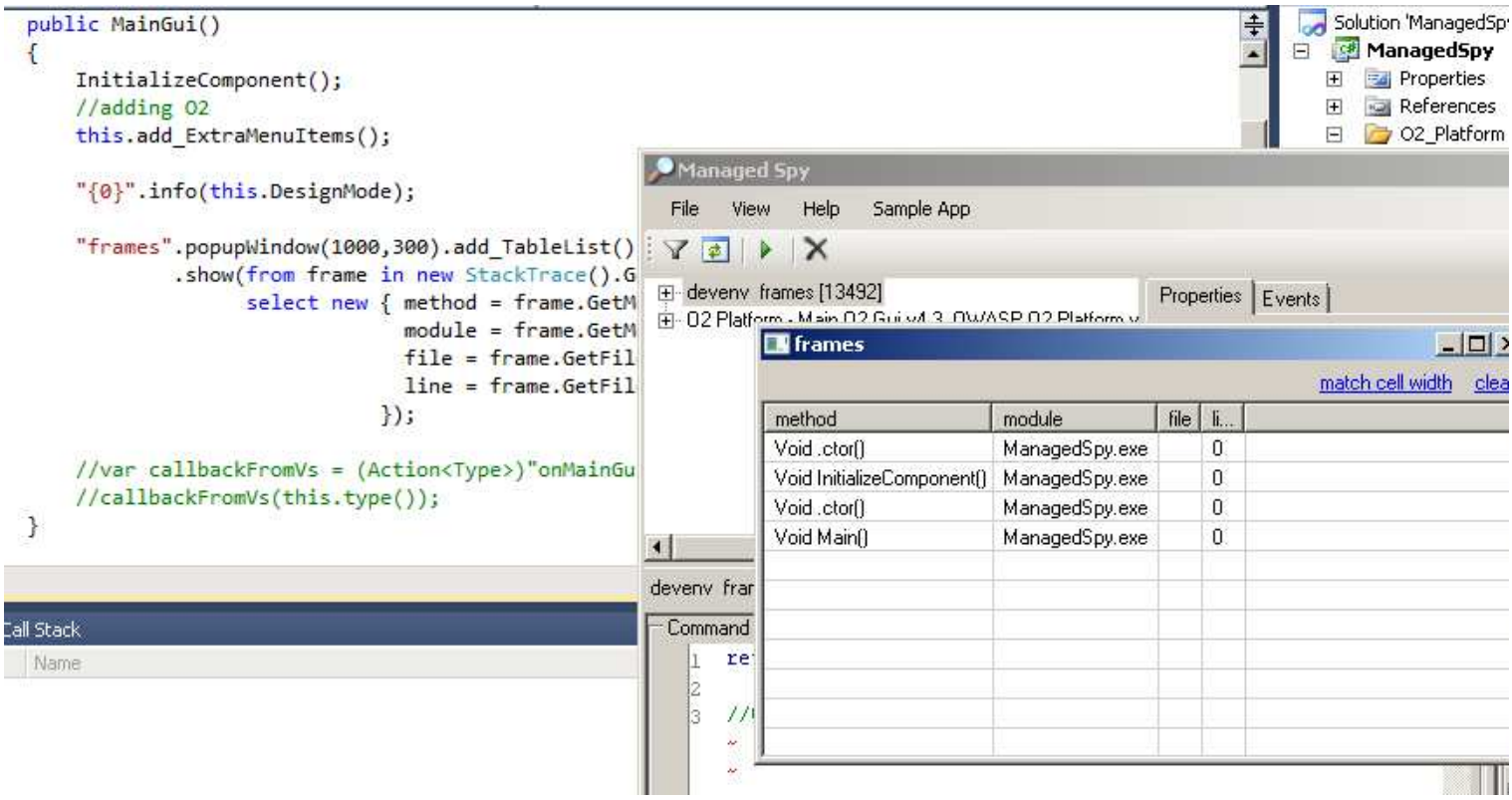
We will get a very different StackTrace

frames

match cell width cle

method	module
Void <dynamicMethod>b__0()	tmp7531.tmp._o2_Script.dll
Void <dynamicMethod>b__2(System.Type)	tmp7531.tmp._o2_Script.dll
Void .ctor()	ManagedSpy.exe
System.Object CreateInstance(System.RuntimeType, Boolean, Boolean, Boolean ByRef, System.RuntimeMethod...	CommonLanguageRuntimeLibrary
System.Object CreateInstanceSlow(Boolean, Boolean, Boolean)	CommonLanguageRuntimeLibrary
System.Object CreateInstanceDefaultCtor(Boolean, Boolean, Boolean, Boolean)	CommonLanguageRuntimeLibrary
System.Object CreateInstance(System.Type, Boolean)	CommonLanguageRuntimeLibrary
System.Object CreateInstance(System.Type)	CommonLanguageRuntimeLibrary
System.Object createObjectUsingDefaultConstructor(System.Type)	O2_FluentSharp_CoreLib.dll
System.Windows.Forms.Control <add_Control>b__3()	O2_FluentSharp_BCL.dll
T invokeThread[T](System.Func`1[T])	O2_FluentSharp_BCL.dll
Void <invokeOnThread>b__0(System.Object, System.EventArgs)	O2_FluentSharp_BCL.dll
Void InvokeMarshaledCallbackDo(ThreadMethodEntry)	System.Windows.Forms.dll
Void InvokeMarshaledCallbackHelper(System.Object)	System.Windows.Forms.dll
Void runTryCode(System.Object)	CommonLanguageRuntimeLibrary
Void ExecuteCodeWithGuaranteedCleanup(TryCode, CleanupCode, System.Object)	CommonLanguageRuntimeLibrary
Void RunInternal(System.Threading.ExecutionContext, System.Threading.ContextCallback, System.Object)	CommonLanguageRuntimeLibrary
Void Run(System.Threading.ExecutionContext, System.Threading.ContextCallback, System.Object, Boolean)	CommonLanguageRuntimeLibrary
Void Run(System.Threading.ExecutionContext, System.Threading.ContextCallback, System.Object)	CommonLanguageRuntimeLibrary
Void InvokeMarshaledCallback(ThreadMethodEntry)	System.Windows.Forms.dll
Void InvokeMarshaledCallbacks()	System.Windows.Forms.dll
Void WndProc(System.Windows.Forms.Message ByRef)	System.Windows.Forms.dll
Void WndProc(System.Windows.Forms.Message ByRef)	System.Windows.Forms.dll
Void OnMessage(System.Windows.Forms.Message ByRef)	System.Windows.Forms.dll
Void WndProc(System.Windows.Forms.Message ByRef)	System.Windows.Forms.dll
IntPtr DebuggableCallback(IntPtr, Int32, IntPtr, IntPtr)	System.Windows.Forms.dll
IntPtr DispatchMessageW(MSG ByRef)	System.Windows.Forms.dll
Boolean System.Windows.Forms.UnsafeNativeMethods.IMsoComponentManager.FPushMessageLoop(IntPtr, Int...	System.Windows.Forms.dll
Void RunMessageLoopInner(Int32, System.Windows.Forms.ApplicationContext)	System.Windows.Forms.dll
Void RunMessageLoop(Int32, System.Windows.Forms.ApplicationContext)	System.Windows.Forms.dll
Void RunDialog(System.Windows.Forms.Form)	System.Windows.Forms.dll
System.Windows.Forms.DialogResult ShowDialog(System.Windows.Forms.IWin32Window)	System.Windows.Forms.dll
System.Windows.Forms.DialogResult ShowDialog()	System.Windows.Forms.dll
Void showDialog(Boolean)	O2_FluentSharp_BCL.dll
Void <showAscxInForm>b__0()	O2_FluentSharp_BCL.dll
Void <staThread>b__0()	O2_FluentSharp_CoreLib.dll
Void ThreadStart_Context(System.Object)	CommonLanguageRuntimeLibrary
Void Run(System.Threading.ExecutionContext, System.Threading.ContextCallback, System.Object, Boolean)	CommonLanguageRuntimeLibrary
Void Run(System.Threading.ExecutionContext, System.Threading.ContextCallback, System.Object)	CommonLanguageRuntimeLibrary
Void ThreadStart()	CommonLanguageRuntimeLibrary

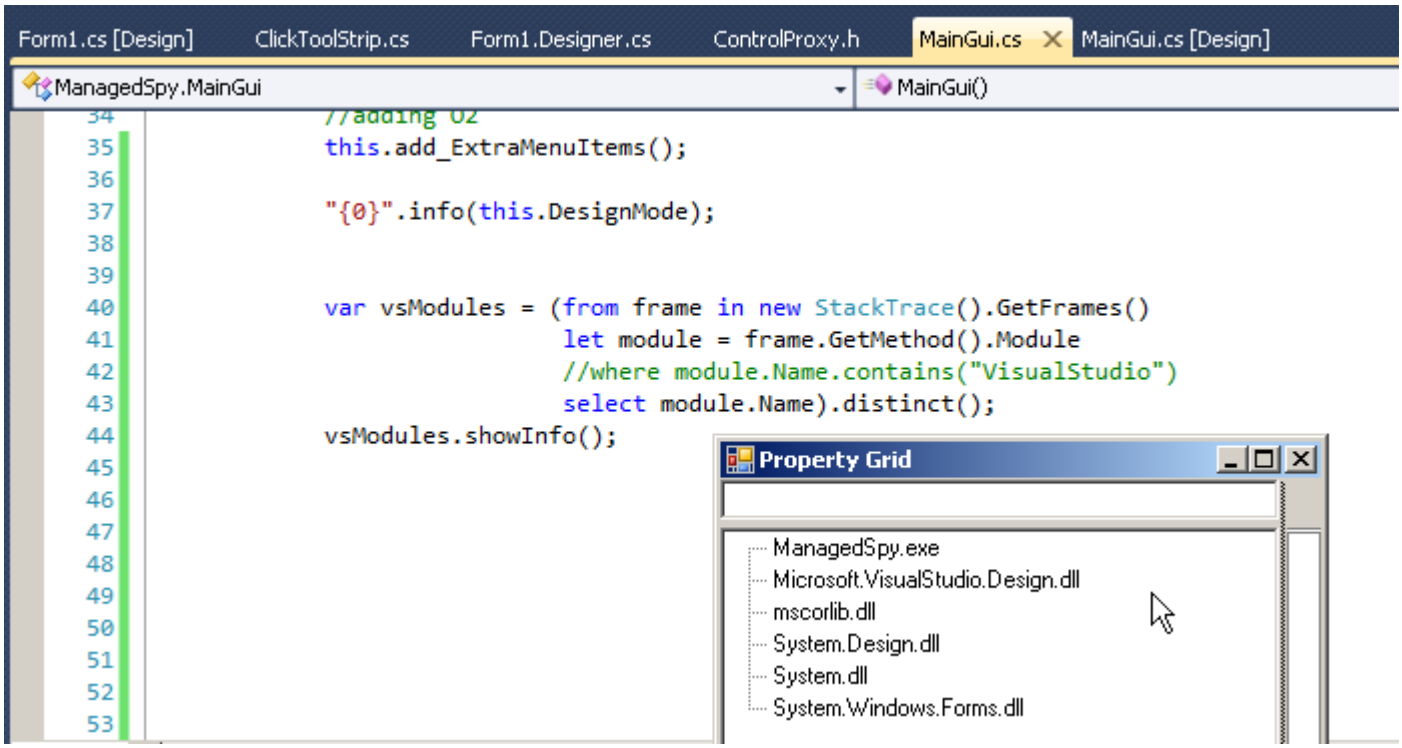
for reference here is the StackTrace when the actually ManagedLib.exe is executed:



so if we look at the list of Modules in the StackTrace:

```
var vsModules = (from frame in new StackTrace().GetFrames()
                 let module = frame.GetMethod().Module
                 //where module.Name.contains("VisualStudio")
                 select module.Name).distinct();

vsModules.showInfo();
```

We will see a reference to a VisualStudio dll



which we can use to detect if the callback is from VisualStudio Designer (or from O2's PopupWindow or the main

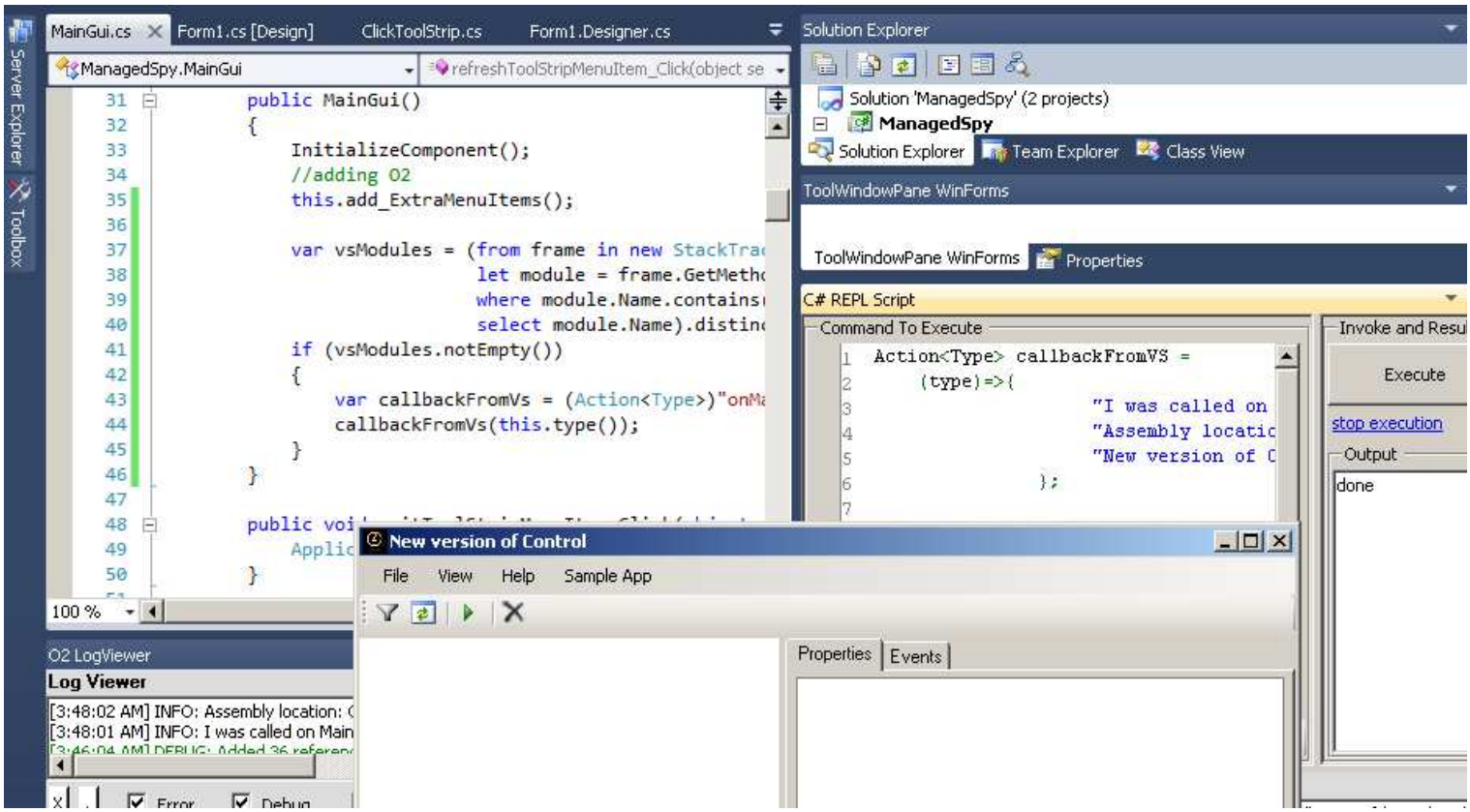
ManagedSpy.exe execution)

```
public MainGui()
{
    InitializeComponent();
    //adding O2
    this.add_ExtraMenuItems();

    var vsModules = (from frame in new StackTrace().GetFrames()
                     let module = frame.GetMethod().Module
                     where module.Name.contains("VisualStudio")
                     select module.Name).distinct();

    if (vsModules.notEmpty())
    {
        var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cache();
        callbackFromVs(this.type());
    }
}
```

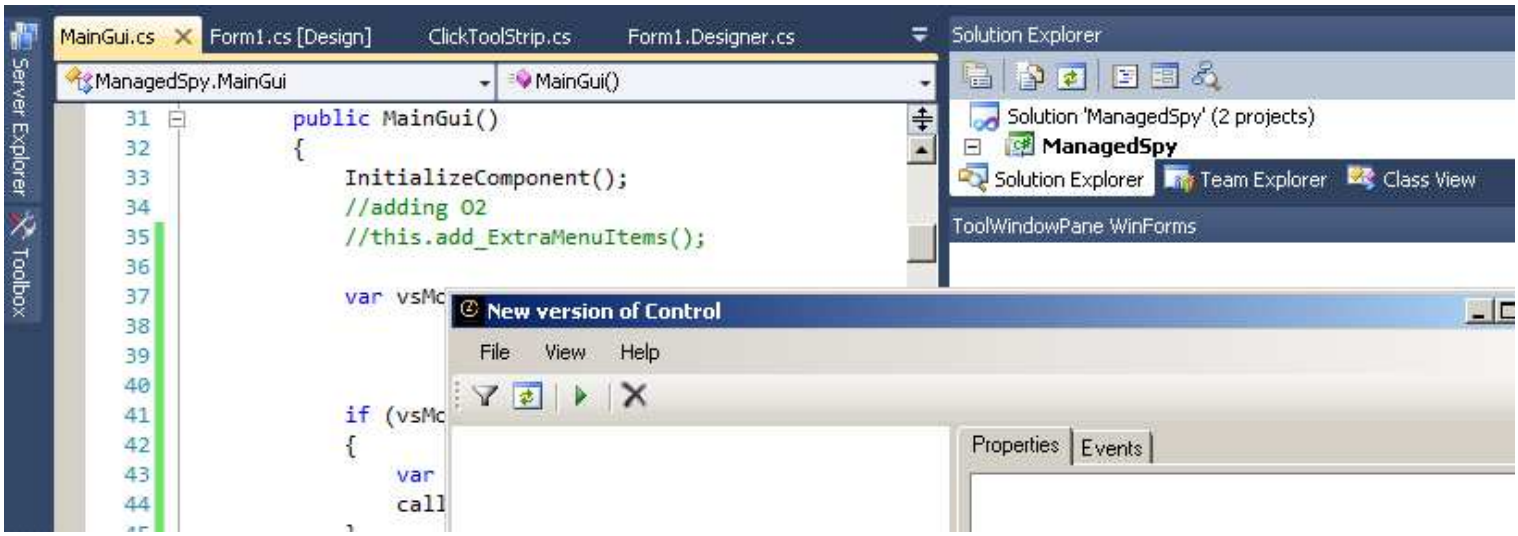
with the callback looking like this:

```
Action<Type> callbackFromVS =
      (type)=>{
                  "I was called on MainGui ctor with: {0}".info(type);
                  "Assembly location: {0}".info(type.assemblyLocation());
                  "New version of Control".popupWindow().add_Control(type);
            };


"onMainGuiCtor".o2Cache(callbackFromVS);

return "done";
//using System.Diagnostics
```

And on project build we will get a (one) popup window with the latest version of the MainGui control



To confirm that we are really seeing the latest version, lets comment the *ExtraMenuItems* (and on next build notice how they are not there)

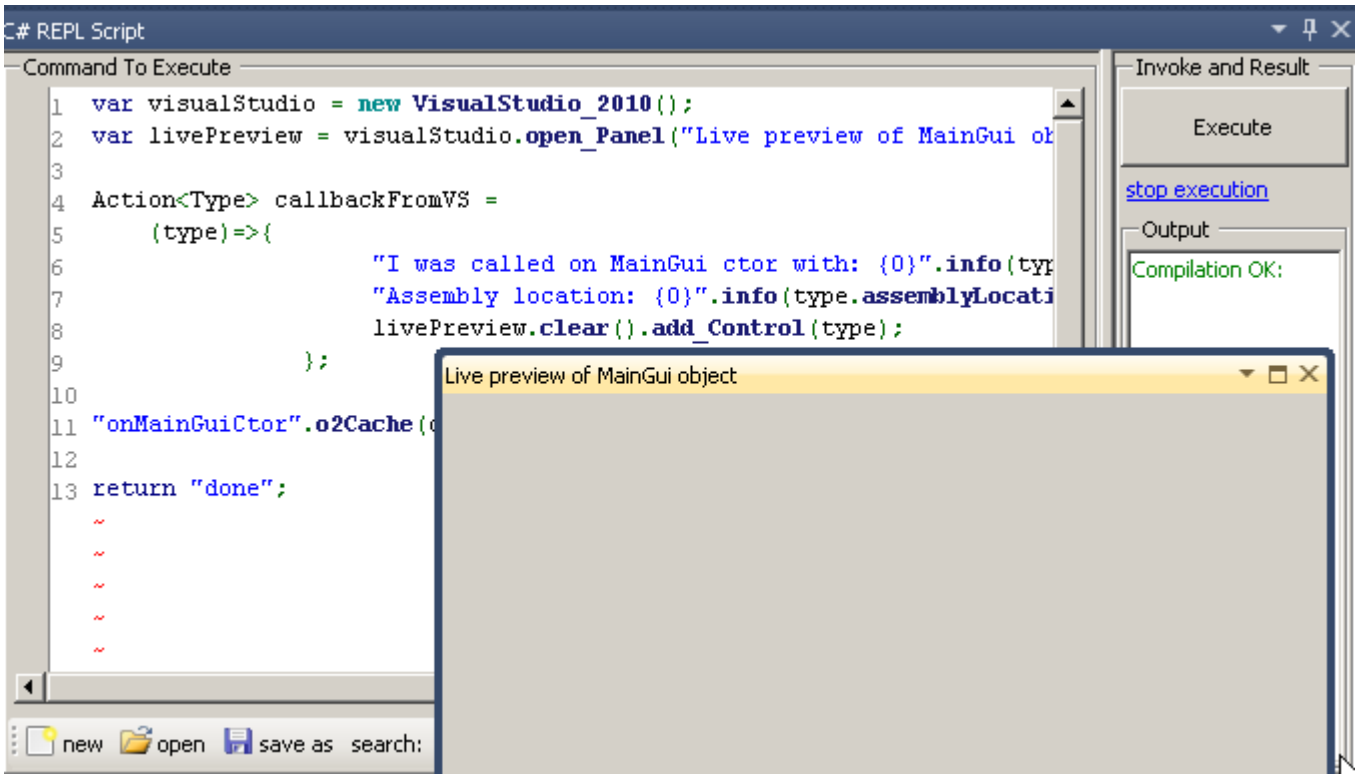


Now at the moment the new control is being shown on a standard WinForms Form Control, which is not native to VisualStudio and can't be docked or tabbed

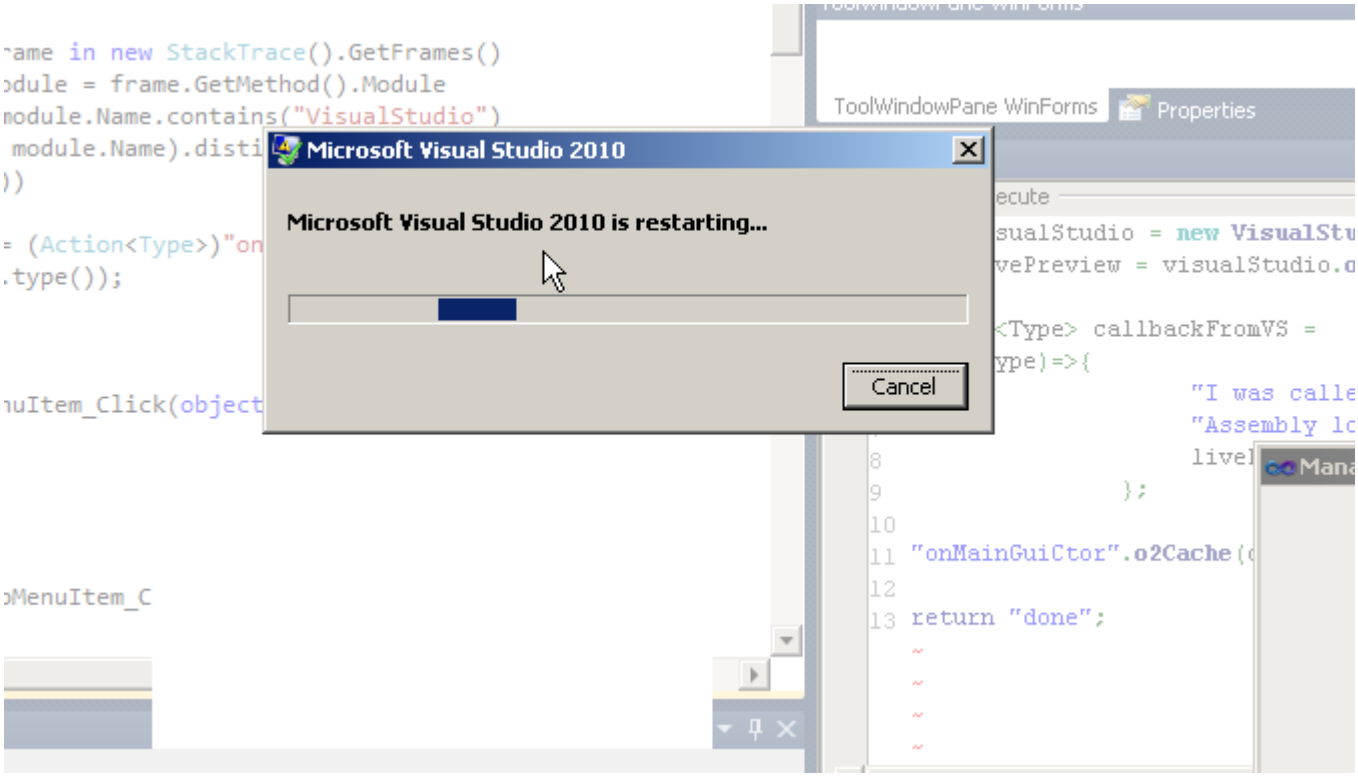
So lets create a VisualStudio window with a panel (which can then be used to add the latest instance of the MainGui control)

```
var visualStudio = new VisualStudio_2010();
var livePreview = visualStudio.open_Panel("Live preview of MainGui object");

Action<Type> callbackFromVS =
       (type)=>{
                                    "I was called on MainGui ctor with: {0}".info(type);
                                    "Assembly location: {0}".info(type.assemblyLocation());

                                    livePreview.clear().add_Control(type);
                        };

"onMainGuiCtor".o2Cache(callbackFromVS);

return "done";
```

when executed, we will have a new VS Window called *'Live preview of MainGui object'*



The last code crashed VS:



Which is most likely due to a threading issues, so let's create the control from an MtaThread:

```
var visualStudio = new VisualStudio_2010();
var livePreview = visualStudio.open_Panel("Live preview of MainGui object");

Action<Type> callbackFromVS =
       (type)=>{
                      "I was called on MainGui ctor with: {0}".info(type);
                      "Assembly location: {0}".info(type.assemblyLocation());
                      O2Thread.mtaThread(
                             ()=>{
                                    livePreview.clear().add_Control(type);
                                 });
                      };

"onMainGuiCtor".o2Cache(callbackFromVS);

return "done";
```

And now on Compile we get the latest version of the MainGui control created in the *'Live preview of MainGui object'* VS Window

Form1.cs [Design] Form1.Designer.cs ControlProxy.h MainGui.cs MainGui.cs [Design]

ManagedSpy.MainGui MainGui()

```
30
31          public MainGui()
32          {
33              InitializeComponent();
34              //adding O2
35              //this.add_ExtraMenuItems();
36
37              var vsModules =
38
39
40
41              if (vsModules.n
42              {
43                  var callbac
44                  callbackFro
45              }
46          }
47
48          public void exitToo
49              Application.Exi
50          }
51
52
53
54          public void refresh
```

Live preview of MainGui object

File View Help

Properties Events

Which can be docked and used to see the latest changes on build (for example let's add back the *ExtraMenuItems*)

MainGui.cs Form1.cs [Design] Form1.Designer.cs ControlProxy.h MainGui.cs [Design]

ManagedSpy.MainGui MainGui()

```
30
31          public MainGui()
32          {
33              InitializeComponent();
34              //adding O2
35              this.add_ExtraMenuItems();
36
37              var vsModules = (from frame in new StackTrace().GetFrames()
38                               let module = frame.GetMethod().Module
39                              where module.Name.contains("VisualStudio")
40                              select module.Name).distinct();
41              if (vsModules.notEmpty())
42              {
43                  var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cache();
44                  callbackFromVs(this.type());
45              }
46          }
47
48          public void exitToolStripMenuItem_Click(object sender, EventArgs e) {
49              Application.Exit();
50          }
51
52
53
54          public void refreshToolStripMenuItem_Click(object sender, EventArgs e)
```

100 %

Live preview of MainGui object

File View Help Sample App

Properties Events

,

Recaping:

The code to execute (once) on the C# REPL window to setup the environment is:

```
var visualStudio = new VisualStudio_2010();
var livePreview = visualStudio.open_Panel("Live preview of MainGui object");

Action<Type> callbackFromVS =
       (type)=>{
                                    "I was called on MainGui ctor with: {0}".info(type);
                                    "Assembly location: {0}".info(type.assemblyLocation());

                                    O2Thread.mtaThread(
                                           ()=>{
                                                         livePreview.clear().add_Control(type);

                                                  });
                      };

"onMainGuiCtor".o2Cache(callbackFromVS);

return "done";
```

The code to add to the UserControl Ctor (to trigger its creation) is:

```
          public MainGui()
        {
            InitializeComponent();
            //adding O2
            this.add_ExtraMenuItems();

            var vsModules = (from frame in new StackTrace().GetFrames()
                              let module = frame.GetMethod().Module
                             where module.Name.contains("VisualStudio")
                             select module.Name).distinct();
            if (vsModules.notEmpty())
            {
                var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cache();
                callbackFromVs(this.type());
            }
        }
```

with both creating an environment where the a live instance of the UserControl is shown (without needing to leave VisualStudio and start the ManagedLib.exe application)

```
public void exitToolStripMenuItem_Click(object sender, EventArgs e) {
    Application.Exit();
}



public void refreshToolStripMenuItem_Click(object sender, EventArgs e) {
    RefreshWindows();
}

/// <summary>
/// This rebuilds the window hierarchy
/// </summary>
public void RefreshWindows() {
```

```
var visualStudio = new VisualStudio_2010();
var livePreview = visualStudio.open_Panel("Live preview of MainGui ol

Action<Type> callbackFromVS =
    (type)=>{
                "I was called on MainGui ctor with: {0}".info(ty
                "Assembly location: {0}".info(type.assemblyLocat:
                O2Thread.mtaThread(
                    ()=>{
                        livePreview.clear().add_Control(type
```

Environment where it is possible to dynamically manipulate the live instance (in the example below, we are adding a new treenode, showing the properties of the current control, and making all child controls pink)

```
{
    InitializeComponent();
    //adding O2
    this.add_ExtraMenuItems();

    this.treeWindow.add_Node("test");
    this.propertyGrid.show(this);
    this.controls(true).pink();

    var vsModules = (from frame in new StackTrace().GetFrames()
                     let module = frame.GetMethod().Module
                     where module.Name.contains("VisualStudio")
                     select module.Name).distinct();
    if (vsModules.notEmpty())
    {
        var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cach
        callbackFromVs(this.type());
    }
}
```

Let do a last refectoring and to move the VS Callback and Object creation into an extension method:

```
10  {
11      public static class MainGui_ExtensionMethods
12      {
13          public static string TestFile1 { get; set; }
14
15          static MainGui_ExtensionMethods()...
19
20          public static MainGui add_ExtraMenuItems(this MainGui mainGui)...
28
29          public static MainGui enableVisualStudioObjectCreation(this MainGui mainGui)
30          {
31              var vsModules = (from frame in new StackTrace().GetFrames()
32                                 let module = frame.GetMethod().Module
33                                where module.Name.contains("VisualStudio")
34                                select module.Name).distinct();
35              if (vsModules.notEmpty())
36              {
37                  var callbackFromVs = (Action<Type>)"onMainGuiCtor".o2Cache();
38                  callbackFromVs(mainGui.type());
39              }
40              return mainGui;
41          }
```

Which will simplify the MainGui ctor code

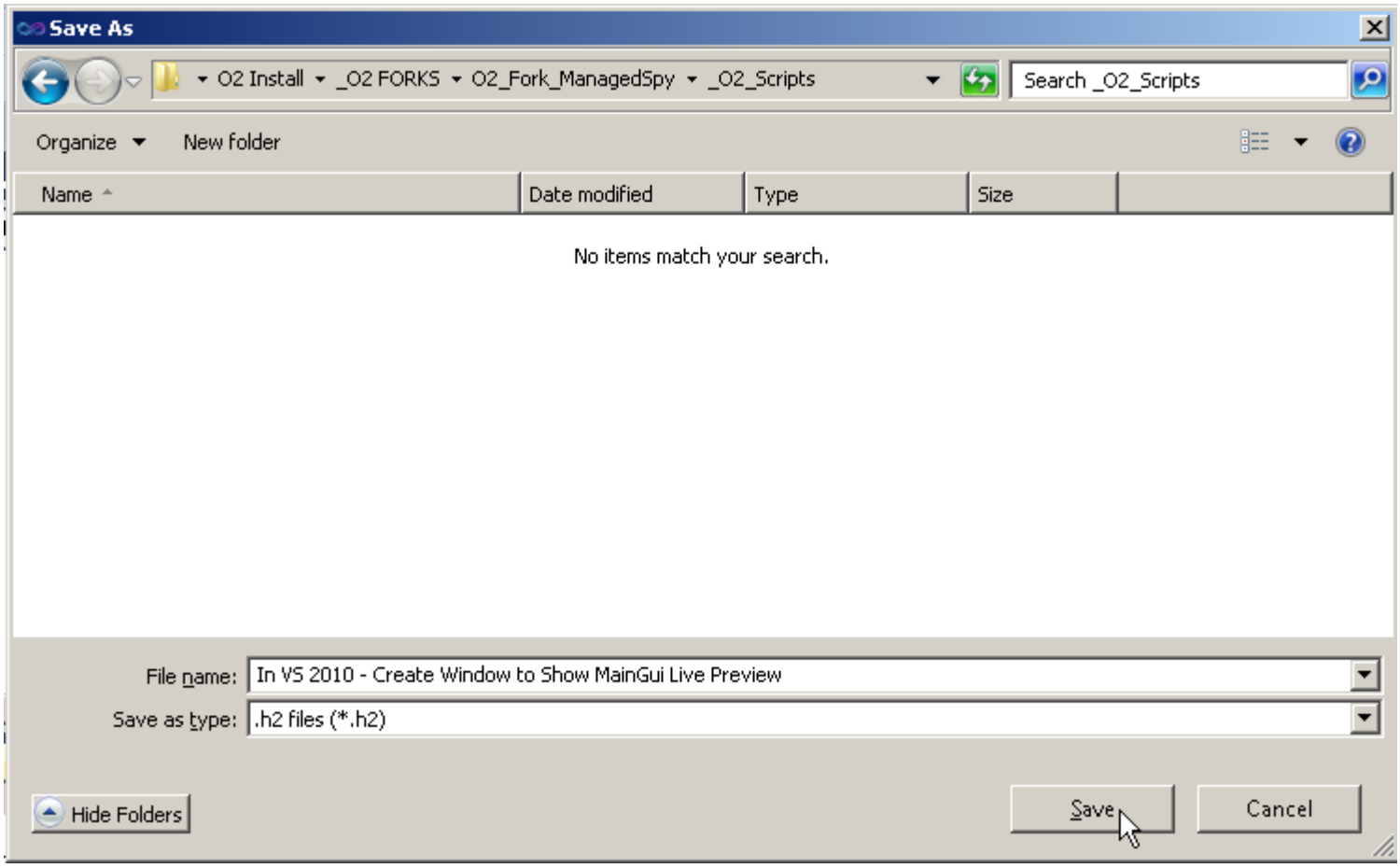
```
MainGui_ExtensionMethods.cs   MainGui.cs   Form1.cs [Design]   Form1.Designer.cs
ManagedSpy.MainGui                          MainGui()
30
31          public MainGui()
32          {
33              InitializeComponent();
34              this.enableVisualStudioObjectCreation();
35
36              //adding O2
37              this.add_ExtraMenuItems();
38
39              this.treeWindow.add_Node("this was created..")
40                              .add_Node("inside VisualStudio");
41
42
43          }
```

Live preview of MainGui object

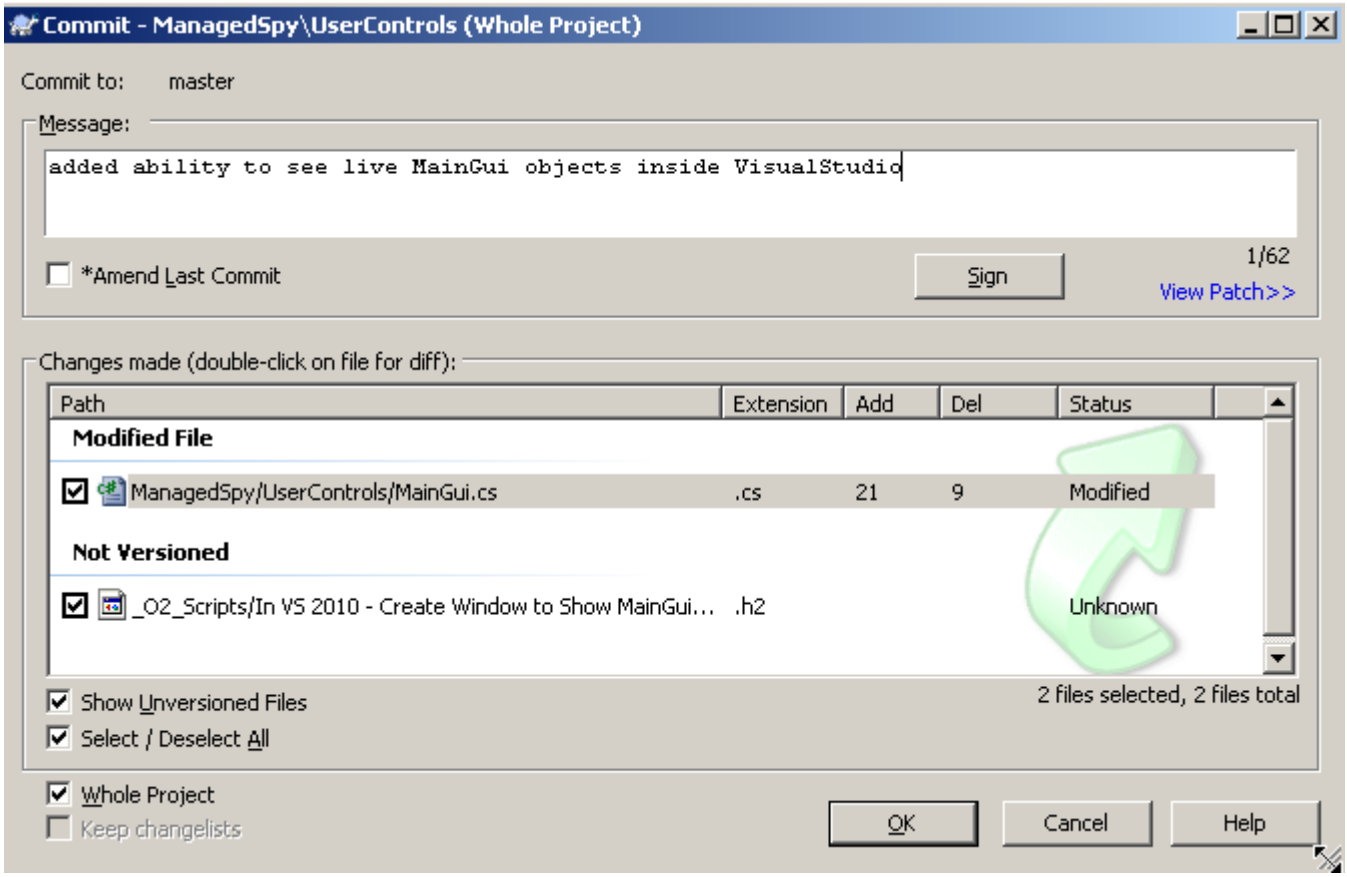
File View Help Sample

this was created..

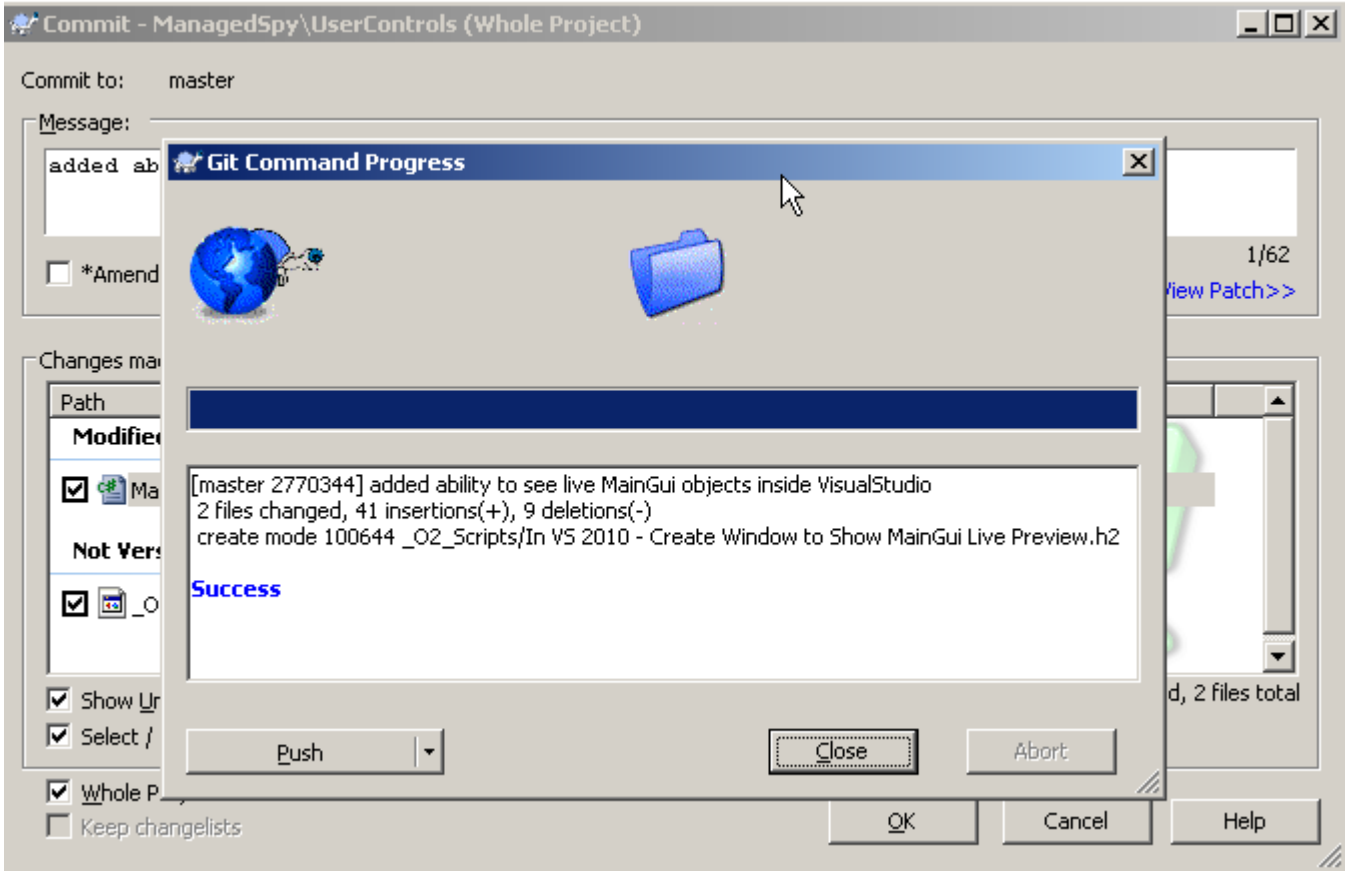
Finally, to recreate this environment later, the C# REPL script was saved as an *.h2 script:



with the changes:



commited as:


Commit - ManagedSpy\UserControls (Whole Project)
Commit to: master
Message:
added ab
*Amend
1/62
View Patch>>
Changes ma
Path
Modifie
Ma
Not Vers
_O
Show Ur
Select /
d, 2 files total
Whole P
Keep changelists
OK
Cancel
Help
Git Command Progress
[master 2770344] added ability to see live MainGui objects inside VisualStudio
2 files changed, 41 insertions(+), 9 deletions(-)
create mode 100644 _O2_Scripts/In VS 2010 - Create Window to Show MainGui Live Preview.h2
Success
Push
Close
Abort