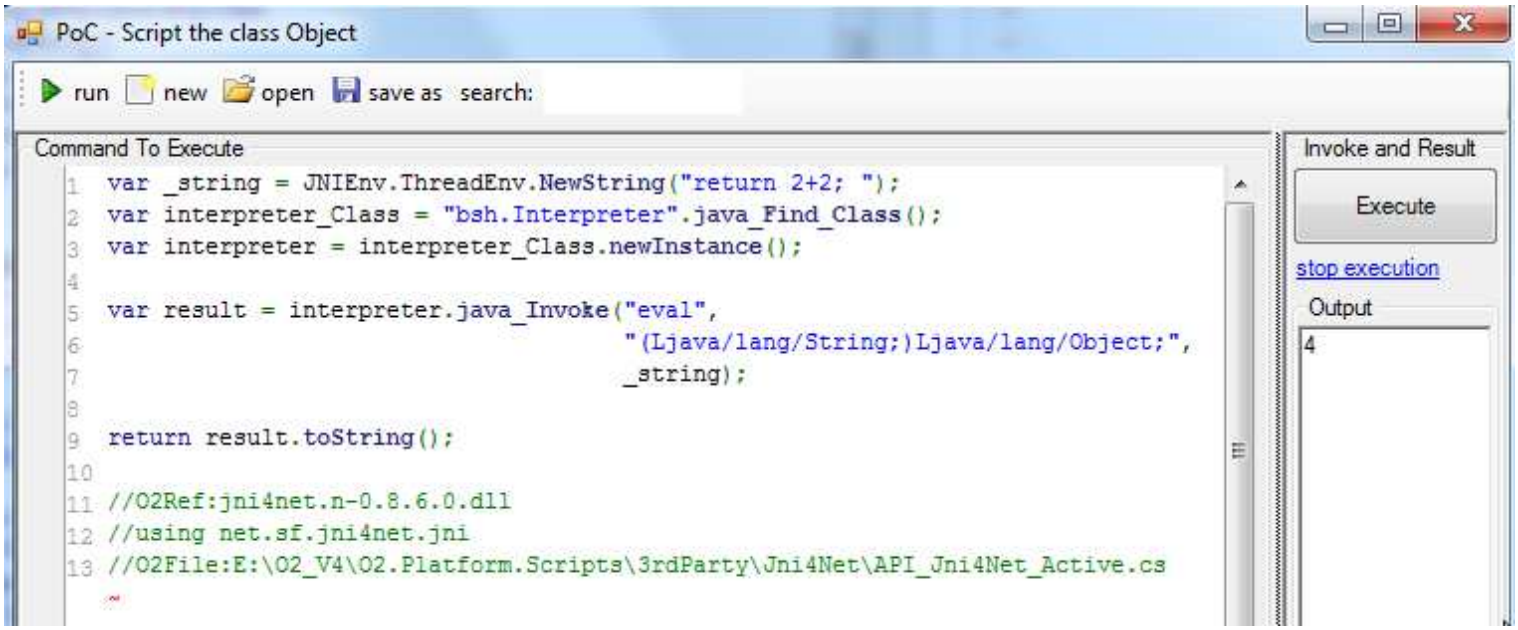
Using Jni4Net (Part 2) - Controling OWASP ZAP remotely (via Java BeanShell REPL in .Net)

see previous document: **using Jni4Net to C# REPL a java process (ZAP Proxy)** where the last script showed how to:

Invoking BeanShell from C# REPL

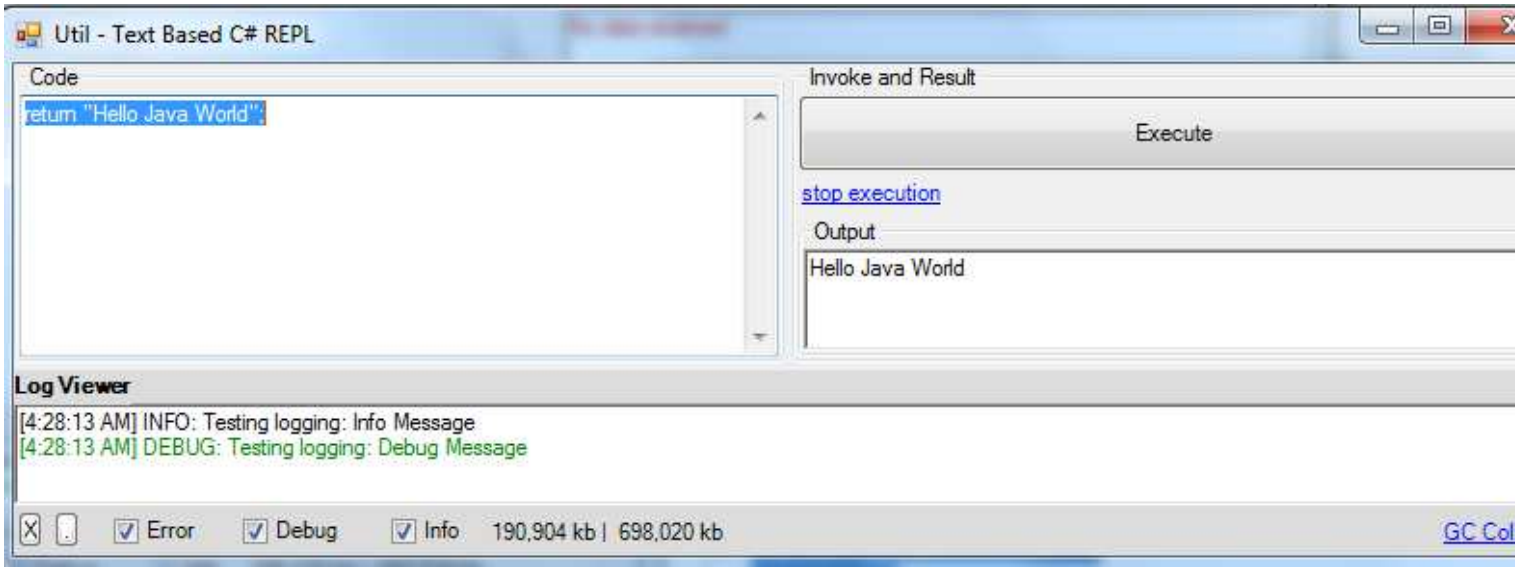
```
var _string = JNIEnv.ThreadEnv.NewString("return 2+2; ");
var interpreter_Class = "bsh.Interpreter".java_Find_Class();
var interpreter = interpreter_Class.newInstance();

var result = interpreter.java_Invoke("eval",
                                     "(Ljava/lang/String;)Ljava/lang/Object;",
                                    _string);

return result.toString();

//O2Ref:jni4net.n-0.8.6.0.dll
//using net.sf.jni4net.jni
//O2File:E:\O2_V4\O2.Platform.Scripts\3rdParty\Jni4Net\API_Jni4Net_Active.cs
```



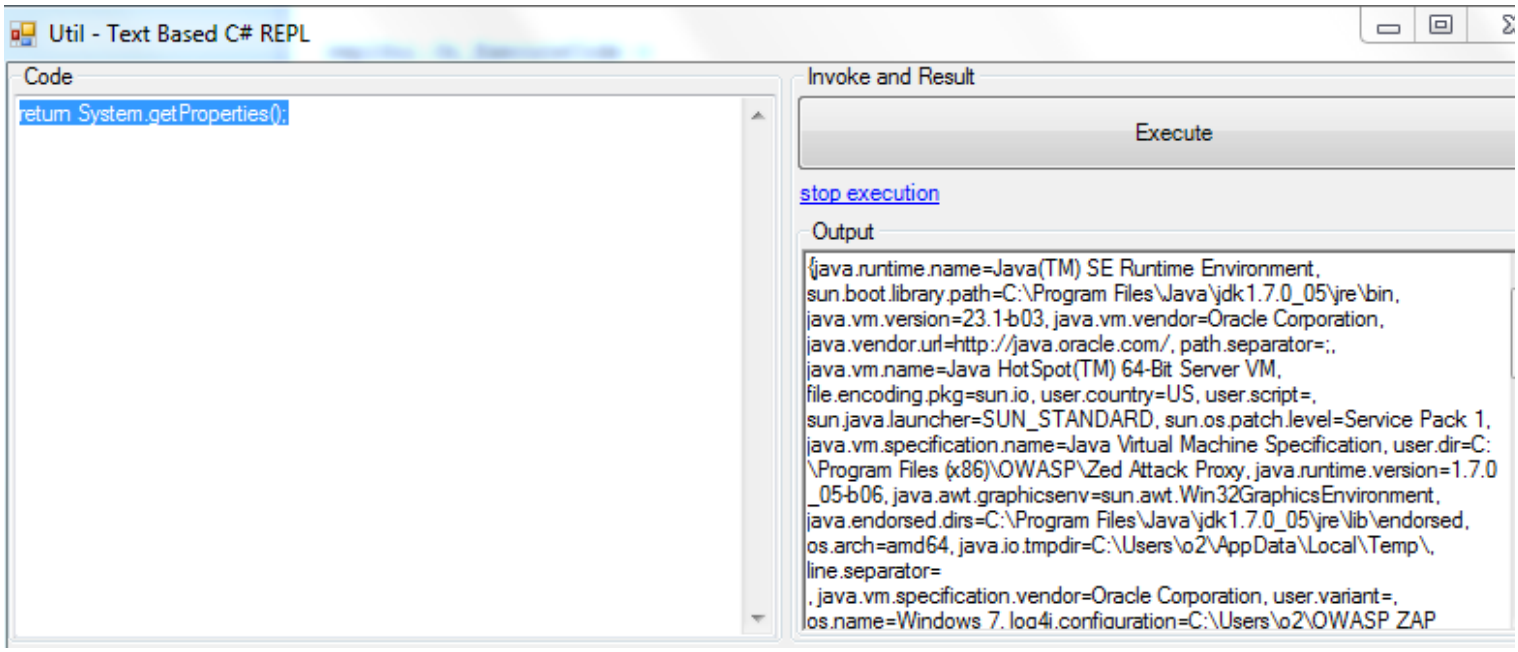
BeanShell REPL GUI (code in CLR with invoke happing in JVM)

```
var topPanel = "Util - Text Based C# REPL".popupWindow(800,400).insert_LogViewer();
//var topPanel = panel.clear().add_Panel();

var replGui = topPanel.add_REPL_Gui();

var codeText = replGui.Code_Panel.add_TextArea().allowTabs();

var interpreter_Class = "bsh.Interpreter".java_Find_Class();
var interpreter = interpreter_Class.newInstance();

replGui.On_ExecuteCode =
      ()=>{
            var snippet = JNIEnv.ThreadEnv.NewString(codeText.get_Text());
            var result = interpreter.java_Invoke("eval",
                                                 "(Ljava/lang/String;)Ljava/lang/Object;",
                                                snippet);
            if (result.notNull())
                  replGui.showOutput(result.toString());
            else
                  replGui.showErrorMessage("No data received from JVM invoke");
      };

codeText.set_Text("return \"Hello Java World\";");
replGui.On_ExecuteCode();
```

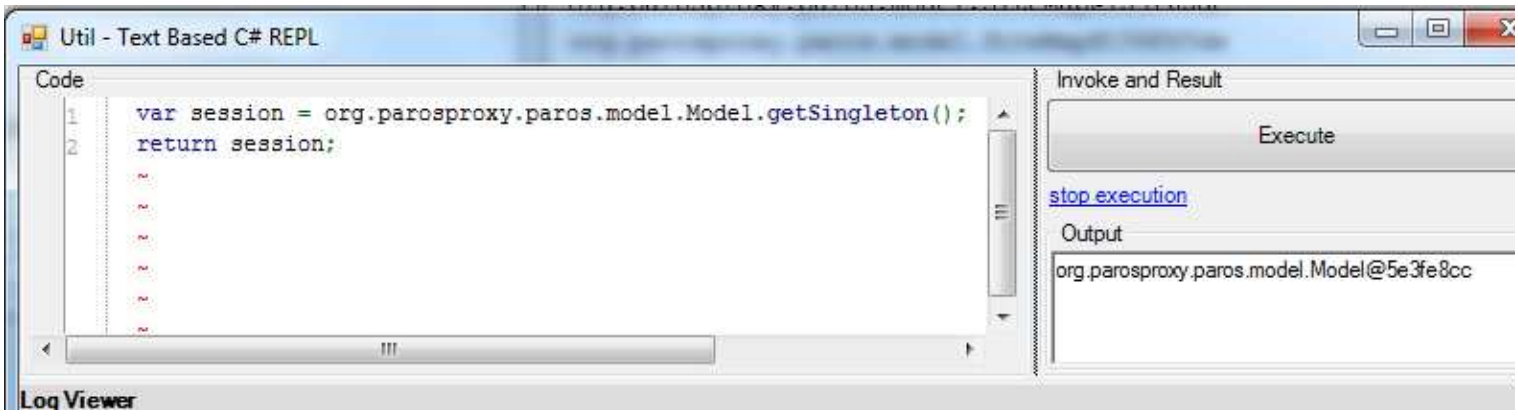
```
//O2Ref:jni4net.n-0.8.6.0.dll
//using net.sf.jni4net.jni
//O2File:E:\O2_V4\O2.Platform.Scripts\3rdParty\Jni4Net\API_Jni4Net_Active.cs
```



Getting JVM properties (with code compiled and executed via the BeanShell Interactive class):

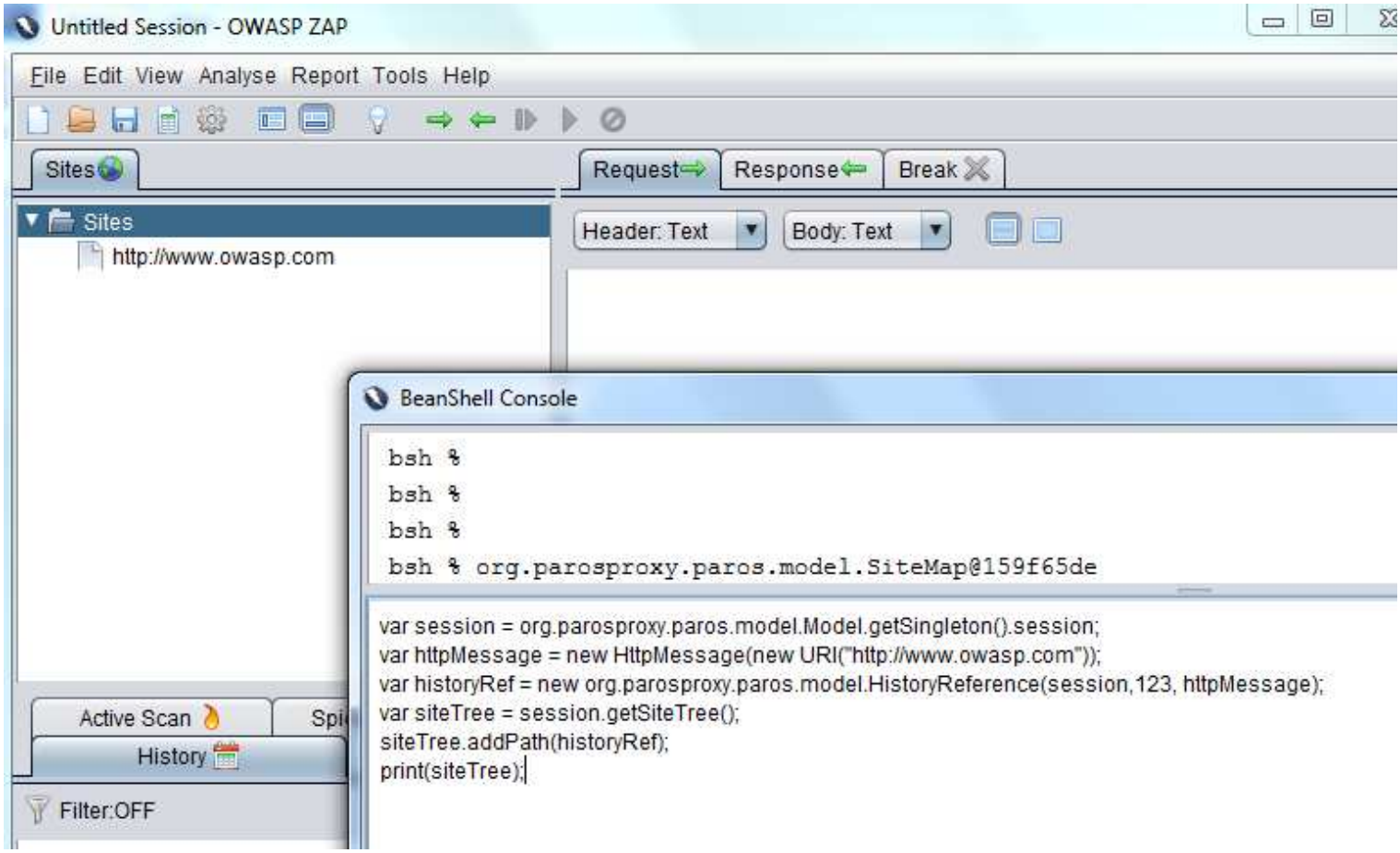


Getting Zap project session (changed text editor to be a SourceCodeEditor)



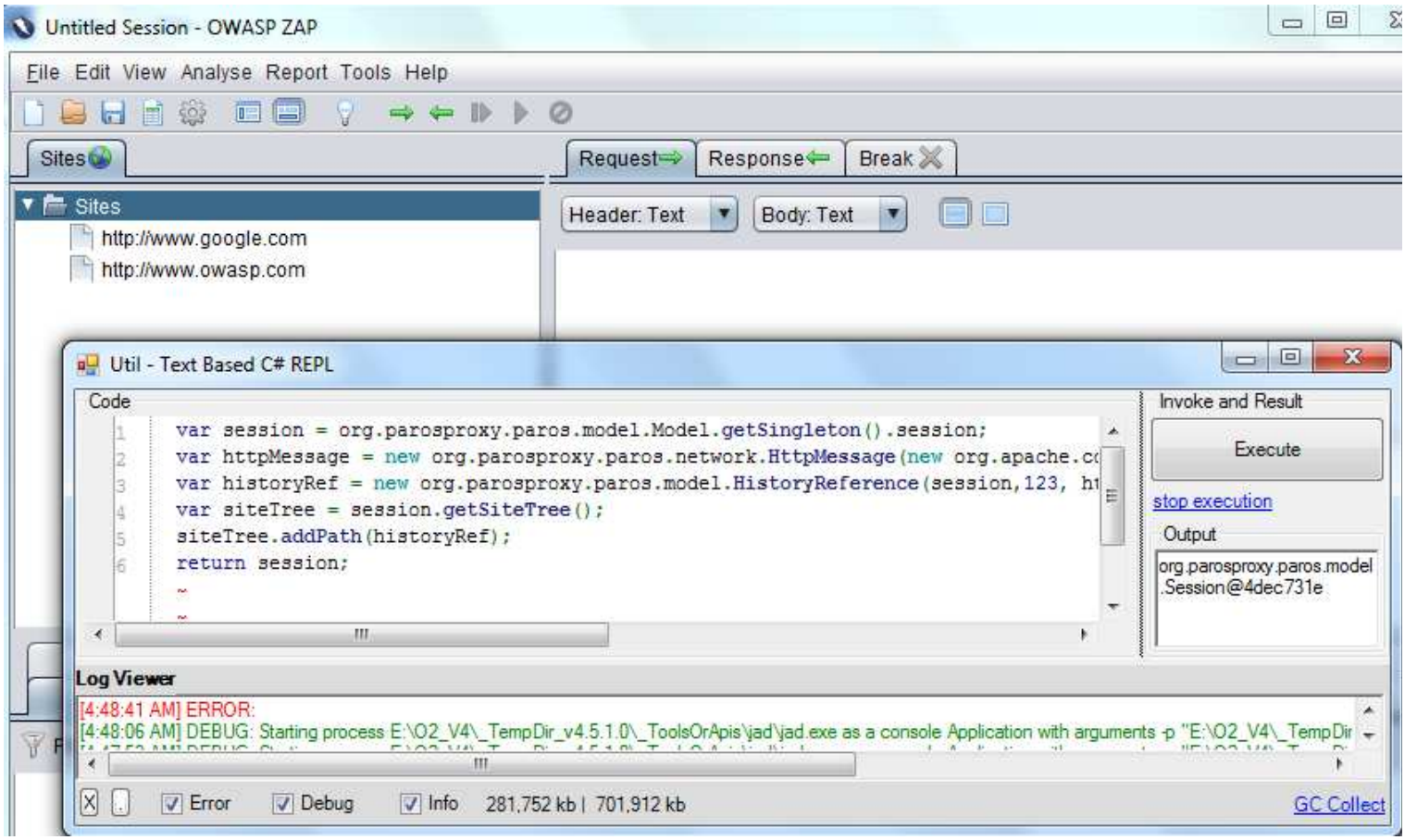
BeanShell script to add new *Path* to the *SiteTree* (executed from ZAP BeanShell console)

```
var session = org.parosproxy.paros.model.Model.getSingleton().session;
var httpMessage = new HttpMessage(new URI("http://www.owasp.com"));
var historyRef = new org.parosproxy.paros.model.HistoryReference(session,123, httpMessage);
var siteTree = session.getSiteTree();
siteTree.addPath(historyRef);

print(siteTree);
```



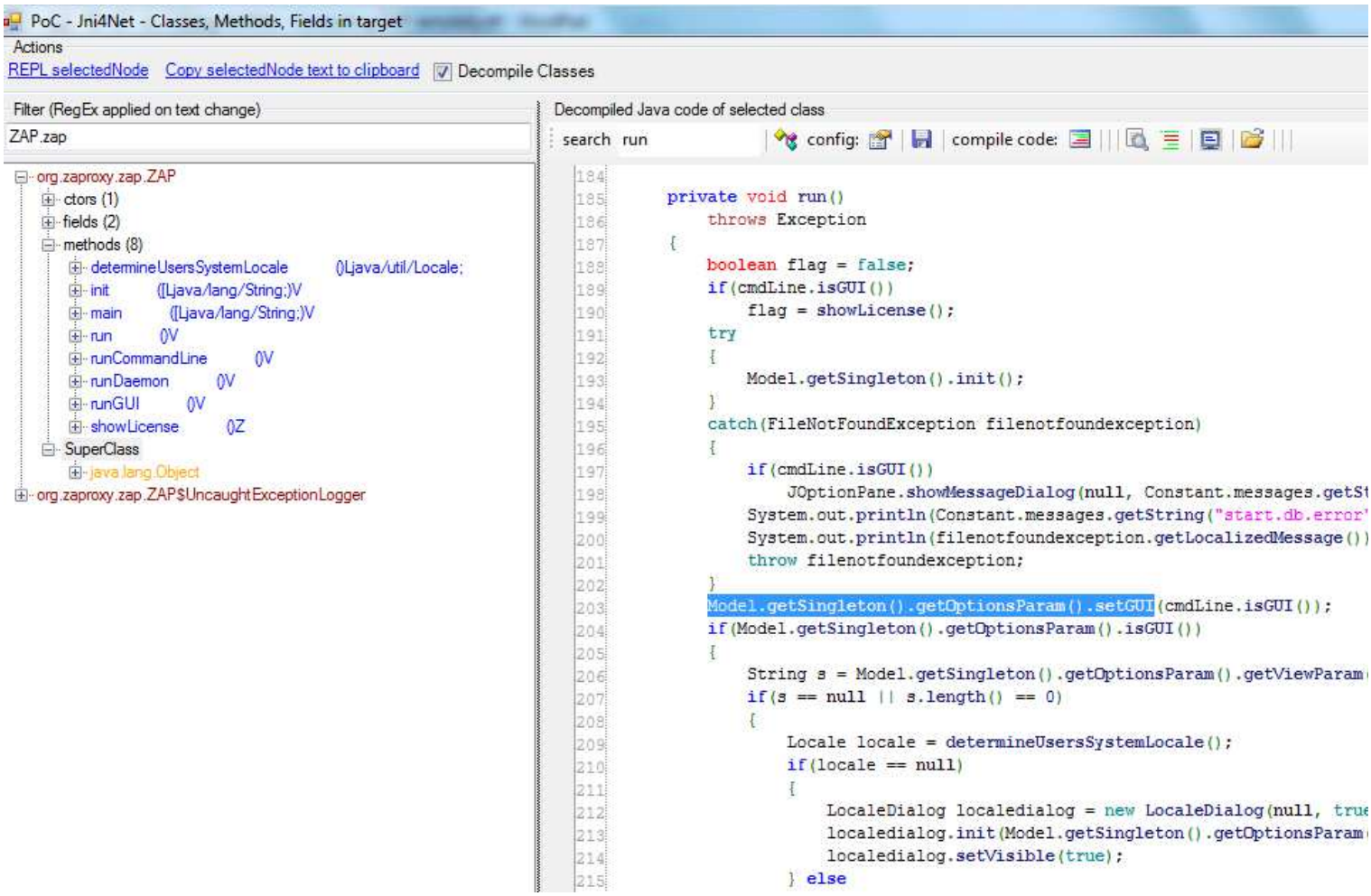
BeanShell script to add new *Path* to the *SiteTree* (executed from C# Repl GUI (using method's FullName)

```
var session = org.parosproxy.paros.model.Model.getSingleton().session;
var httpMessage = new org.parosproxy.paros.network.HttpMessage(new org.apache.commons.httpclient.URI
("http://www.google.com"));
var historyRef = new org.parosproxy.paros.model.HistoryReference(session,123, httpMessage);
var siteTree = session.getSiteTree();
siteTree.addPath(historyRef);
return session;
```



(this same action could be done via CLR-side reflecion (on JVM side objects)

Looking at the source code of the ZAP class methods (like main and run)



showing or hiding the main ZAP window:

```
var control = org.parosproxy.paros.control.Control.getSingleton();
var view = org.parosproxy.paros.view.View.getSingleton();
var mainFrame = view.getMainFrame();
mainFrame.setVisible(false);
```

.

.

.

.

.

.

.

.

.