LS2J – Calling Java From LotusScript

By Julian Robichaux, http://www.nsftools.com
originally published on lotususergroup.org, May 2008

In Notes 6, Lotus introduced a feature called LS2J that allows you to call Java classes from LotusScript. Even though it's a feature that's been around for a while, it doesn't seem to be very widely used. Why? Don't see the point? Performance fears? Lack of examples?

Let's take a look at LS2J and see where it might be useful.

WHY USE JAVA AT ALL?

The first question in the list is often: why use Java in the first place? What's wrong with just plain LotusScript?

Answer: there's nothing wrong with LotusScript. I still use it every day for most of my Lotus Notes programming needs. However, Java can have some significant advantages, including:

1. Network sockets. Java can easily and natively perform platform-independent network communications – HTTP requests, FTP transfers, pings, etc.
2. Image manipulation. There is good built-in support for cross-platform image file manipulation in Java, as well as a number of third-party libraries that enhance the native functionality.
3. Threads. You don't often have to use threads in your programs, but when you need them, you need them. Java allows you to spawn new threads very easily.
4. Libraries and sample code. Even though blogs and websites like OpenNTF have made LotusScript examples more accessible, the enormous amount of Java examples available on the Internet is staggering. And there are Java code libraries out there that do almost anything you want.

That's just the tip of the iceberg. There are all sorts of things you can do in Java that are difficult if not impossible to do in LotusScript (certainly LotusScript can call external DLLs to do many of these things, but those are Windows-only and often require installation on client and server machines).

In addition, the new Notes 8 Standard client has a number of extension points (like Composite Applications and Sidebar Apps) that you can only take advantage of with Java programming.

The beauty of LS2J is that you can combine the power of Java with the ease and familiarity of

LotusScript. The syntax might be a little tricky at first, but once you get used to it you'll find that it's quite easy to mix your Java and your LotusScript code and libraries together.

WHAT ABOUT PERFORMANCE?

A common fear about using LS2J is that the performance is bad or there are stability issues.

Certainly, there are some bugs that have been fixed since the early days of LS2J. You should also be very explicit about freeing Java resources that are called from LS2J, because Java doesn't "clean itself up" as well as LotusScript does. In addition, you'll notice that the very first time you launch an LS2J agent on a workstation, it may take several seconds to start up, due to the fact that the Java JVM usually needs to be initialized before the first run of an LS2J agent.

However, the performance of Java and LS2J itself (after the JVM is up) is comparable to regular LotusScript, and is sometimes even faster for some operations like large String manipulation. Don't be afraid of running it because it's slow. Yes, there is a certain amount of overhead involved with going back and forth between LotusScript and Java, but it's very unlikely that you'd notice such a thing in real-world user testing.

As with everything, test like crazy before deploying into production. With Java and LS2J I don't tend to worry about performance as much as I do about memory usage (yes, those are two different things). As with any programming language, there are things you can do in Java/ LS2J that could cripple or crash your client or server – likewise, I can crash a client or server with LotusScript in about 3 lines of poorly written code. Despite that, there are very impressive things you can do with Java and LS2J that are worth the effort of testing and debugging.

HOW CAN I USE IT?

Here's some example LS2J code so you can see what it looks like:

```
Option Public
Option Explicit
Uselsx "*javacon"                '** required for LS2J
Use "JpgImage Library"           '** Java Script Library we'll use
Sub Initialize
    Dim jSession As New JavaSession
    Dim jpgClass As JavaClass
    Dim jpgImage As JavaObject
    Dim fileName As String, newFileName As String
```

```
    fileName = "C:\test.jpg"            '** we'll modify this JPEG file
    Set jpgClass = jSession.GetClass("JpgImage")
    Set jpgImage = jpgClass.CreateObject("(Ljava/lang/String;)V", fileName)
    Call jpgImage.scalePercent(0.5)         '** shrink to 50% of the original size
    Call jpgImage.grayscale()               '** make it black and white
    newFileName = jpgImage.sendToFile(jpgImage, "new-" & fileName)
    Print fileName & " has been modified and saved as " & newFileName
End Sub
```

This code calls a custom “JpgImage” Java class from the “JpgImage Library” script library (which is a Java library) and uses it to modify a JPG image file. For the most part, using the object that was created from the Java class is very similar (if not identical) to using a native LotusScript class object. You have access to the class methods, and you pass parameters and access return values the same way.

If a Java method or constructor is “overloaded” (meaning that there are different versions of the same method with different parameters – a nice feature that Java has), you have to use some unusual syntax, like we see in the CreateObject line with the “(Ljava/lang/String;)V” parameter. There is information on this in the Domino Designer help.

The big advantage to an LS2J agent like this over a LotusScript agent that calls a DLL is: all the code required is in the database, so there are no external dependencies required. Also, the code is completely cross-platform, so it will work on a Windows client as well as a Linux/Unix server. (N.B.: Macintosh clients didn't get Java support until just recently.)

So, there you go. Another fun thing to play with if you're not familiar with it already. Also, here are some examples to get you started:

http://www.nsftools.com/tips/NotesTips.htm#ls2jexamples

http://nsftools.com/tips/NotesTips.htm#zipfilemanager

http://www.convergens.dk/C1256BB3004BB983/0/AAB236D45A9E7CE1C1256F3F002CFE2D

http://www.thenorth.com/APBLOG4.nsf/Threaded/D1AAF414346C43CB85256F00004B3365