

LiveConnect

Kevin Andrusky
Stephen Bodnar
Darshan Gill

Overview

- Introduction
- JavaScript to Java
- Java to JavaScript
- Controlling Plug-ins
- Plug-ins to Java
- Plug-ins to JavaScript

Introduction

- Developed by Netscape to permit Communication between:
 - Java
 - JavaScript, and
 - Plug-ins
- Implemented in Netscape 3.0
- Later implemented by Microsoft for Internet Explorer 4.0

JavaScript to Java

Controlling Applets

Accessing Applets:

- 1) document.applets[appletindex]
- 2) document.applets['applet']
- 3) document.applet

Where:

```
<APPLET CODE="RandomCircles.class" WIDTH=100 HEIGHT=100  
NAME="applet">  
</APPLET>
```

We can call the `startdrawing()` method in the applet as follows:

```
document.applets['applet'].startdrawing(); or  
document.applet.startdrawing();
```

Calling Java Methods

Call methods using:

Packages.<packagename>.<classname>.<methodname>

Packages in `netscape.*`, `java.*`, and `sun.*` do not require Packages prefix

Example:

```
var date = new Packages.java.util.Date();  
System.out.println(date);
```

or

```
var date = new java.util.Date();  
System.out.println(date);
```

Type Conversions

JavaScript

Java

number

float

boolean

boolean

string

String

Other

JSObject

Type Conversions

Objects that are wrappers around Java objects are unwrapped.

Other objects are wrapped with a JSObject.

This means that all JavaScript values appear in Java as objects of class `java.lang.Object`. To use them, you generally will have to cast them to the appropriate subclass of `Object`, for example:

```
(String) window.getMember("name");  
(JSObject) window.getMember("document");
```

Java to JavaScript

Preparation

To prepare to compile Java applets to use LiveConnect:
Add the path to `java40.jar` to your CLASSPATH.
Use `import` to import the packages in `netscape.javascript.*`:
`import netscape.javascript.*;`

To enable Java applets to connect to JavaScript use the `MAYSCRIPT` property of the `APPLET` tag:
`<APPLET CODE="RandomCircles.class" WIDTH=100 HEIGHT=100 NAME="applet" MAYSCRIPT>`
`</APPLET>`

The `MAYSCRIPT` attribute prevents an applet from accessing JavaScript on a page without the knowledge of the page author

JSObject API (part)

<code>getWindow(Applet a)</code>	Retrieves the window handle.	
<code>getMember(String name)</code>	Retrieves a named member of a JavaScript object.	<code>this.name</code>
<code>setMember(String name, Object value)</code>	Sets a named member of a JavaScript object.	<code>this.name = value</code>
<code>call(String methodName, Object args[])</code>	Calls a JavaScript method.	<code>This.methodName (arg[0] ...)</code>

Example

```
//get the window handle for the window containing this applet
JSObject window = JSObject.getWindow(this);
```

```
//get the document from the DOM
JSObject doc = (JSObject) window.getMember("document");
```

```
//work with elements of the document
JSObject form = (JSObject) doc.getMember("theform");
JSObject textfield = (JSObject) form.getMember("text");
String st = (String) textfield.getMember("value");
textfield.setMember("value", "Hi, I've just changed you");
form.call("submit", null);
```

Type Conversions

Java	Javascript
Numeric types	Number
Boolean	Boolean
JSObject	Original Object
Array	Array
Other	Java Object Wrapper

Controlling Plug-ins

Prerequisites

Controlling from Applets and JavaScript

The plug-in must be a subclass of `netscape.plugin.Plugin`. Then all Java methods declared public in the Plug-in are available to Java applets.

If the plug-in is not a subclass of `netscape.plugin.Plugin`, then we cannot access any of its methods.

For Applets

For an applet to access a plug-in, it must have the property `MAYSCRIPT`

Controlling Plug-ins

From Java:

```
PluginSubClassName pl = (PluginSubClassName)  
doc.getMember("pluginname");
```

This gives you access to the methods defined in the subclass.

From JavaScript:

- 1) `document.embed[embedindex]`
- 2) `document.embed['plugin']`
- 3) `document.plugin`

Where: `<EMBED SRC="..." NAME="plugin" ...></EMBED>`

We can call the `startdrawing()` method in the plug-in as follows:

```
document.embed['plugin'].startdrawing(); or  
document.plugin.startdrawing();
```

Plug-ins to Java

- Complicated
 - Involves combining C and Java
- See:
<http://www.netscape.com/eng/mozilla/3.0/handbook/plugins/pjava.htm>

Plug-ins to JavaScript

- If Plug-in is a subclass of `netscape.plugin.Plugin`
 - Same as Java to JavaScript methods

Conclusions

- Powerful technology, though not widely used.
- Rather antiquated, Netscape 6 doesn't have full support for LiveConnect anymore
- Newer technologies?
 - Flash?

LiveConnect Examples

- Java To JavaScript:
 - <http://www.apl.jhu.edu/~hall/java/JavaScript-from-Java.html>
- JavaScript To Java:
 - <http://www.netscape.com/eng/mozilla/3.0/handbook/javascript/livecon.htm>
 - <http://www.apl.jhu.edu/~hall/java/Java-from-JavaScript.html>
 - <http://www.webreference.com/javascript/961202/part01.html>

LiveConnect Info

References We Used

- <http://www.netscape.com/eng/mozilla/3.0/handbook/javascript/livecon.htm>
- <http://www.netscape.com/eng/mozilla/3.0/handbook/plugins/>
- <http://developer.netscape.com/docs/manuals/communicator/jsref/pkg.htm>
- http://www.js-examples.com/javascript/core_js15/lc.php3
- <http://developer.netscape.com/docs/manuals/signedobj/trust/owp.htm>
- <http://java.sun.com/j2se/1.3/docs/guide/plugin/security.html>