Embedded to External Web Browser Transition Tips

These tips will help you transition existing solutions designed to work with the Maya embedded web browser to the external web browser approach (Maya 8.0).
We provide a JavaScript file that contains many useful Maya-web browser communication functions, including browser and platform detection, presence of plug-in, and error handling.
See the MayaWebTool.js file in the <Maya location>/ExternalWebBrowser/Examples directory.

Initialization

Old code:
With the embedded web browser, connection between Maya and JavaScript was handled by the XPCOM component.

```
var imel;
netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect");
imel = Components.classes["@alias.com/CommandEngine/MEL;1"].getService();
imel = imel.QueryInterface(Components.interfaces.nsIMEL);
```

New code for Firefox, Safari, etc.:
In browsers which support the Netscape Plugin API with scriptability extension (e.g., Mozilla, Firefox, Safari), the connection between Maya and JavaScript is handled by the browser plugin. The plugin must be installed for your browser, and in order to use it from a web page, an instance of plugin must be loaded.

```
<embed name="mcpPlugin" type="application/x-mcp" hidden="true"></embed>
```

New code for Internet Explorer on Windows:
Internet explorer uses a COM object to handle the connection to Maya. The COM object must be registered in your operating system.

(JavaScript or Visual Basic scripting languages can be used in your web documents.)

```
Dim MEL
Set MEL = CreateObject("Maya.CommandEngine.MEL")
```

Connecting to the Maya Command Port

Code for Firefox, Safari, etc.:
The browser plugin has a *port* property. Setting the *port* property value to Maya Command Port name establishes a connection to this port.

```
var MEL = document.embeds[0]
MEL.port = "myCommandPortName";
```

Code for Internet Explorer on Windows:

The COM object has a *PortName* property. In order to connect to the Maya Command Port, you have to set this property and then call the Connect() method.

```
MEL.PortName = "myCommandPortName"
MEL.Connect()
```

Executing MEL commands

Old code:

```
imel.ExecuteCommand("sphere", 1, 1);
```

New code for Firefox, Safari, etc.:

```
MEL.execute('sphere')
```

New code for Internet Explorer on Windows:

```
MEL.Execute("sphere")
```

Getting results

Old code:

In the old code, you would call one of the methods to get results based on the type. For example, a string array:

```
var array;
var count = {};
array = imel.GetStringArrayResult(count);
```

New code for Firefox, Safari, etc.:

The browser plugin returns results from Maya when you send a command to execute using the "execute" method. Getting a result is just a simple assignment of the method call to the JavaScript object.

```
var res = MEL.execute('sphere')
// Array results returned as a tab-delimited strings
var resArray = res.split('\t')
```

New code for Internet Explorer on Windows:

When using the COM object to get the result of last executed command, you have to query the "Result" property.

```
MEL.Execute("sphere")
Dim resArray
resArray = MEL.Result
```

Disconnecting from Maya Command Port

Code for Firefox, Safari, etc.:

The browser plugin has a *port* property. Setting the *port* property value to an empty string closes the connection to the Maya Command Port.

```
MEL.port = "";
```

Code for Internet Explorer on Windows:

The COM object has a Disconnect() method.

```
MEL.Disonnect()
```