JavaOne℠

Sun's Worldwide Java Developer Conference

Netscape

Introduction to the JavaScript Language

Netscape: Introduction to the JavaScript Language

Eckart Walther
Netscape Communications

Serving Up:

- JavaScript Overview
- Server-side JavaScript
- LiveConnect: Integrating JavaScript, Java Applets & Native Code
- Future Directions

What Is JavaScript?

"JavaScript is an object-based scripting language designed to add programmatic capabilities and cross platform scripting of events, objects, and methods to web pages."

Java vs. JavaScript

JavaScript	Java
Interpreted (not compiled) by client.	Compiled on server before execution on client.
Object-based. Code uses built-in, extensible objects, but no classes or inheritance.	Object-oriented. Applets cons of object classes with inherita
Code integrated with, and embedded in, HTML.	Applets distinct from HTML (accessed from HTML pages)
Variable data types not declared (loose typing).	Variable data types must be declared (strong typing)
Dynamic binding. Object references checked at run-time.	Static binding. Object referenc must exist at compile-time.

What Does it Taste Like?

- Use <SCRIPT> tag
- Built-in objects, properties extend HTML & reflect document content:
  - Objects: window, frames, document, form, form elements
  - Properties: **document.bgcolor, document.lastModified**
  - Methods: **document.write(), window.open()**
  - Form element event-handlers: **onClick**, **onFocus**

A Simple JavaScript

```
<HTML>
<HEAD>
  <SCRIPT LANGUAGE=“JavaScript”>
  <!-- hide script from old browsers
  document.write( “Welcome to” );
  // end the hiding here -->
  </SCRIPT>
</HEAD>
<BODY>
JavaOne!
</BODY>
</HTML>
```

JavaScript Functions

- Generally defined in HEAD
- Stored and executed by events in the page
-
```
 <HEAD><SCRIPT LANGUAGE=“JavaScript”>
  function square( x ) {
    return x * x;
  }
  document.write( “The function returned “,
                  square( 5 ), “.”);
</SCRIPT></HEAD>
```

Events

```
<TAG eventHandler=“JavaScript Code”>
e.g., <INPUT TYPE=“button”
       VALUE=“Calculate”
       onClick=“compute( this.form )”>
```

- Focus, Blur, Change events: text fields, textareas, and selections
- Click events: buttons, radio buttons, checkboxes, submit buttons, reset buttons, links
- Select events: text fields, textareas
- MouseOver events: links

Functions & Events

```
<HEAD> <SCRIPT LANGUAGE=“JavaScript”>
  function compute( form ) {
    if( confirm( “Are you sure?” ) )
      form.result.value = eval( form.expr.value );
    else
      alert( “Please come back again” );
  }
</SCRIPT> </HEAD>
<BODY>
  <FORM> Enter an expression:
  <INPUT TYPE=“text” NAME=“expr” SIZE=15 >
  <INPUT TYPE=“button” VALUE=“Calculate”
      ONCLICK=“compute( this.form )” >
  <BR> Result:
  <INPUT TYPE=“text” NAME=“result” SIZE=15 >
</BR> </FORM> </BODY>
```

Objects

- Window, Child windows: Top-level object and frame windows
  - window.open(“doc1.html”)
- Location: current URL
  - location.href = "http://www.terrapin.com/samples/vsimple.html"
- History: previous URLs
  - history.length = 7
- Document: title, background color
  - document.myform.Button1.value = “Press Me”

Server-Side JavaScript

- Executes on the server to satisfy an HTTP request
- Includes pre-defined objects
- Includes built-in functions
- Maintains state across requests

HTTP Request: Execution Viewpoint

- 1. Server receives request
- 2. Server checks with JavaScript Run-Time to see if it wants the request
- 3. Server turns the request over to JavaScript
- 4. JavaScript reads a pre-digested form of the HTML page (including JavaScript bytecodes) from a binary file
- 5. JavaScript attaches built-in objects with the JavaScript context
- 6. JavaScript interprets the pre-digested form of the page

HTML with JavaScript: Request Object

```
<HTML>
   <HEAD>
         <TITLE> Hello World </TITLE>
   </HEAD>
   <BODY>
         <H1> Hello World </H1> <P>
         Your IP Address is
         <SERVER>
              write(request.ip)
         </SERVER>
   </BODY>
</HTML>
```

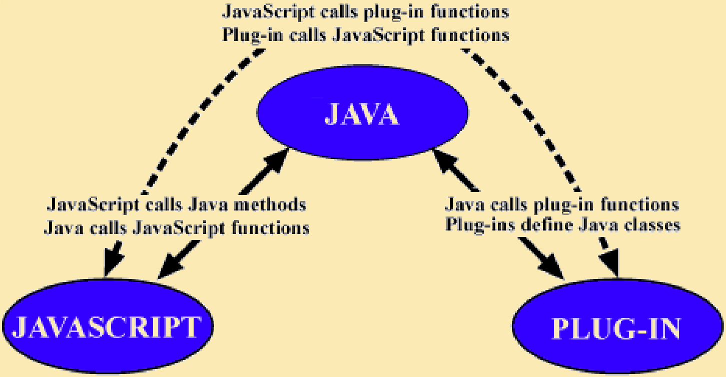
Pre-Defined Objects

- Server
- Project
- Client
- Request
- Database

Build-In Functions

- defined
- write
- flush
- redirect
- debug
- registerCFunction

Connecting All the Pieces



JavaScript calls plug-in functions
Plug-in calls JavaScript functions

JAVA

JavaScript calls Java methods
Java calls JavaScript functions

Java calls plug-in functions
Plug-ins define Java classes

JAVASCRIPT

PLUG-IN

Integrating JavaScript and Java

- JavaScriptFJava:
  - Full reflection of public Java methods and variables in JavaScript

```
<APPLET CODE=Hello.class NAME=helloApplet
 WIDTH=60 HEIGHT=30>
<PARAM NAME=label VALUE=test>
</APPLET>
```

  - You could reference it in JavaScript as follows:
  - `document.helloApplet;`
  - `document.applets["helloApplet"];`

Integrating JavaScript and Java (cont.)

- JavaScriptFJava:
  - Access to Java packages and static information

```
var System = Packages.java.lang.System;
System.err.println("Greetings from
  JavaScript");
```

  - Use the getClass method to access static variables and methods. E.g.,

```
var MyApplet = getClass(document.nervousApplet);
// set a static variable
MyApplet.textSize = 14;
```

Integrating JavaScript and Java (cont.)

- JavaFJavaScript:
- JSObject: JavaScript objects reflected into Java

-
  ```
  import netscape.javascript.*;
  ```

-
  ```
  Object[] args = new Object[1];
  args[0] = new Integer(4);
  JSObject.getWindow( this )
            .call("square", args);
  ```

Integrating JavaScript and Native Components

- Java Runtime Interface (JRI) standard interface to Java services
  - Platform independent native methods
- Calling a Plug-In from JavaScript

```
 <EMBED name=avi WIDTH=150 HEIGHT=150 SRC=
"1204ride.avi">
...
<A HREF="#" onclick="document.avi.play()">
```

Defining Native Methods

- In Java:

```
class AviPlugin extends Plugin {
    public native void play();
}
```

- javah generates C/C++ header file
- In C/C++:

```
void
native_AviPlugin_play(JRIEnv* env,
                      jref plugin) {
    ...
}
```

Future Directions

- Even tighter Integration between JavaScript and Java
- More events, objects, and functions
- Security-Data Tainting
- Integration with JavaBeans framework

Q & A

- More Information about JavaScript
  http://home.netscape.com/eng/mozilla/3.0/handbook/javascript/index.html

JavaOne℠

Sun's Worldwide Java Developer Conference