

The JavaScript Language

Language Elements

- Variables
- Literals
- Operators
- Control Structures
- Functions
- Objects

Javascript Variables

- Untyped!
- Can be declared with var keyword:
`var foo;`
- Can be created automatically by assigning a value:

```
foo=1;      blah="Hi Dave";
```

Variables (cont.)

- Using **var** to declare a variable results in a *local* variable (inside a function).
- If you don't use **var** – the variable is a global variable.

Literals

- The typical bunch:
 - Numbers `17` `123.45`
 - Strings `"Hello Dave"`
 - Boolean: `true` `false`
 - Arrays: `[1, "Hi Dave", 17.234]`



Arrays can hold anything!

Arrays

- We will look at Arrays in more detail a bit later.
- Arrays are actually Javascript Objects.
- The only thing special in the language to support arrays is the syntax for literals...

Operators

- Arithmetic, comparison, assignment, bitwise, boolean (pretty much just like C++).

+ - * / % ++ -- == != > <
&& || ! & | << >>

Different than C++

- The + operator is used for addition (if both operands are numbers)

-or-

- The + operator means string concatenation (if either one of the operands is not a number)

Control Structures

- Again – pretty much just like C:
`if if-else ?: switch`

`for while do-while`
- And a few not in C
`for (var in object)`

`with (object)`

Javascript Functions

- The keyword **function** is used to define a function (subroutine):

```
function add(x,y) {  
    return (x+y) ;  
}
```

- No type is specified for arguments!

Quiz: What is the value of:

<code>add(3, 4)</code>	<code>7</code>
<code>add("3", "4")</code>	<code>7</code>
<code>add("Hi", "Dave")</code>	<code>"HiDave"</code>
<code>add(3, "Hi")</code>	<code>"3Hi"</code>
<code>add("2.13blah", 3.14)</code>	<code>"2.13blah3.14"</code>

Javascript program to make sure

```
<SCRIPT>
function add(x,y) {
    return(x+y);
}

document.write("add(3,4) is " + add(3,4) + "<BR>");
document.write("add(\"3\", \"4\") is " + add("3","4") +
    "<BR>");
document.write("add(\"Hi\", \"Dave\") is " +
    add("Hi","Dave") + "<BR>");
document.write("add(3, \"Hi\") is " + add(3,"Hi") +
    "<BR>");
document.write("add(\"2.13blah\", 3.14) is " +
    add("2.13blah", 3.14));
</SCRIPT>
```

Recursion is supported

```
function factorial(x) {  
    // use <= 0 instead of < 0  
    // to avoid problems with neg numbers  
  
    if (x<=0)  
        return(1) ;  
    else  
        return( x * factorial(x-1)) ;  
}
```

```
document.write("<H3>11! = " +  
    factorial(11) + "</H3>");
```

Objects

- Objects have attributes and methods.
- Many pre-defined objects and object types.
- Using objects follows the syntax of C++/Java:

objectname.attributeName

objectname.methodName()

The **document** object

- Many attributes of the current document are available via the **document** object:

Title	Referrer
URL	Images
Forms	Links
Colors	

document Methods

- `document.write()` like a print statement – the output goes into the HTML document.
- `document.writeln()` adds a newline after printing.

```
document.write("My title is" +  
document.title);
```

Example

```
<HEAD>  
<TITLE>JavaScript is Javalicious</TITLE>  
</HEAD>  
<BODY>  
<H3>I am a web page and here is my  
  name:</H3>  
<SCRIPT>  
document.write(document.title);  
</SCRIPT>  
<HR>
```

The `navigator` Object

- Represents the browser. Read-only!
- Attributes include:

`appName`

`appVersion`

`platform`

← often used to determine
what kind of browser is
being used
(Netscape vs. IE)

navigator Example

```
if (navigator.appName ==  
    "Microsoft Internet Explorer") {  
    document.writeln("<H1>This page  
requires Netscape!</H1>");  
}
```

The **w**indow Object

- Represents the current window.
- There are possible many objects of type **Window**, the predefined object **window** represents the current window.
- Access to, and control of, a number of properties including position and size.

window attributes

- **document**
- **name**
- **status** the status line
- **parent**

some **window** methods

alert()

close()

prompt()

moveTo() **moveBy()**

open()

scroll() **scrollTo()**

resizeBy() **resizeTo()**

The **Math** Object

- Access to mathematical functions and constants.
- Constants: **Math.PI**
- Methods:

Math.abs() , **Math.sin()** ,
Math.log() , **Math.max()** ,
Math.pow() , **Math.random()** ,
Math.sqrt() , ...

Math object in use

```
// returns an integer between 1 and 6
function roll() {
    var x = Math.random();

    // convert to range [0,6.0)
    x = x * 6;
    // add 1 and convert to int
    return parseInt(1+x );
}

document.writeln("Roll is " + roll() );
```

Array Objects

- Arrays are supported as objects.
- Attribute **length**
- Methods include:
concat join pop push reverse sort

Some similarity to C++

- Array indexes start at 0.
- Syntax for accessing an element is the same:

```
a[3]++;  
blah[i] = i*72;
```

New Stuff (different than C++)

- Arrays can grow dynamically – just add new elements at the end.
- Arrays can have *holes*, elements that have no value.
- Array elements can be anything
 - numbers, strings, or arrays!

Creating Array Objects

- With the **new** operator and a size:

```
var x = new Array(10);
```

- With the new operator and an initial set of element values:

```
var y = new Array(18, "hi", 22);
```

- Assignment of an *array literal*

```
var x = [1, 0, 2];
```

Arrays and Loops

```
var a = new Array(4);

for (i=0;i<a.length;i++) {
    a[i]=i;
}

for (j in a) {
    document.writeln(j);
}
```

Another Example

```
var colors = [ "blue",  
              "green",  
              "yellow"] ;  
var x = window.prompt("enter a  
    number") ;  
window.bgColor = colors[x] ;
```

Array of Arrays

- Javascript does not support 2-dimensional arrays (as part of the language).
- BUT – each array element can be an array.
- Resulting syntax looks like C++!

Array of Arrays Example

```
var board = [ [1,2,3],  
              [4,5,6],  
              [7,8,9] ];
```

```
for (i=0;i<3;i++)  
  for (j=0;j<3;j++)  
    board[i][j]++;
```