JavaScript Programming Basic Skills (1)

Wen-Hsiang Lu (盧文祥)
Department of Computer Science and Information Engineering,
National Cheng Kung University
2015/10/30

What is JavaScript?

- JavaScript is a scripting language
  - adds dynamic behavior to web pages
  - changes pages in real-time
  - responds to user events
  - created by Netscape in 1995, evolved from Netscape's LiveScript
  - most commonly used as a client-side language
  - good for small amounts of processing

History

- 1992
  - Oak, Gosling at Sun & FirstPerson
- 1995
  - HotJava
  - LiveScript, Eich at Netscape
- 1996
  - JScript at Microsoft
- 1998
  - ECMAScript (European Computer Manufactures Association)

JavaScript Example 1

- [Ex] part1-2.htm

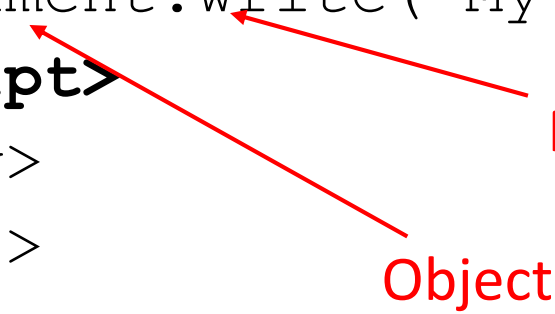
```
<html>
<head>
<title>JavaScript示範</title>
<script language="JavaScript">
    function Welcome(){
      alert("歡迎光臨")
    }
</script>
</head>

<body onload="Welcome()">
JavaScript示範
</body>
</html>
```

Function

User Event

JavaScript Example 2

```
<html>
<head>
<title>JavaScript Statements</title>
</head>
<body>
<script language="JavaScript">
  document.write('My first JavaScript Page');
</script>
</body>
</html>
```



JavaScript Example 2

```
<html>
<head>
<title>JavaScript Statements</title>
</head>
<body>
<script language="JavaScript">
  document.write('<h1>my first JavaScript
  Page</h1>');
</script>
</body>
</html>
```

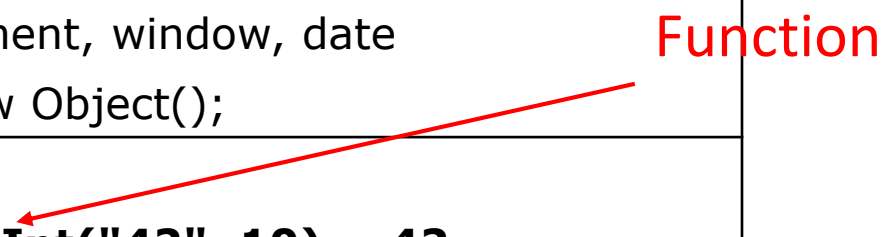
HTML tag written inside JavaScript

Comments in JavaScript

```
<html>
<head>
<title>JavaScript Statements</title>
</head>
<body>
<script language="JavaScript">
   /* print a message by using document object
   and its method write */
   document.write('<h1>This is my first
   JavaScript Page</h1>'); // HTML tag inside JavaScript
</script>
</body>
</html>
```

Data Type

Type	**Example**
Number	x=2 y=3.14
String	x="Happy New Year" y=x+"Mr. Lee!"+"\n"+"Me too."
Boolean	true (1) false (0)
Object	document, window, date x=new Object();
Function	alert **parseInt("42", 10) = 42** user-defined functions



Arithmetic Operators

Operator	Description	Example	Result
+	Addition	x=2 y=2 x+y	4
-	Subtraction	x=5 y=2 x-y	3
*	Multiplication	x=5 y=4 x*y	20
/	Division	15/5 5/2	3 2
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment	x=5 x++	x=6
--	Decrement	x=5 x--	x=4

Assignment Operators

Operator	**Example**	**Is The Same As**
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Comparison Operators

Operator	**Description**	**Example**
==	is equal to	5==8 returns false
!=	is not equal	5!=8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true

Logical Operators

Operator	Description	Example
&&	and	x=6 y=3 (x < 10 && y > 1) returns true
||	or	x=6 y=3 (x==5 || y==5) returns false
!	not	x=6 y=3 !(x==y) returns true

Operator "+"

- Addition and **concatenation**
- If both operands are numbers,
  - then
    - add them: 3 + 4 =7
  - else

    convert them both to strings

    concatenate them

```
'$' + 3 + 4 = '$34'
```

Operator "+"

- Unary operator can convert strings to numbers
  - `+"42" = 42`
- Also
  - `Number("42") = 42`
- Also
  - `parseInt("42", 10) = 42`
  - `+"3" + (+"4") = 7`

String Operations

String Function	Examples	Execution Results
s.charAt(n)	s="abcde"; x=s.charAt(**2**);	x=c
s.indexOf(n)	s="abcde"; x=s.indexOf(c);	x=2
s.length	s="abcde"; x=s.length;	x=5
s.substr(m,n)	s="abcde"; x=s.substr(1,3);	x="bcd"
s.substring(m,n)	s="abcde"; x=s.substring(1,3);	x="bc"
s.concat("string")	s="abcde"; x=s.concat("xyz");	x="abcdexyz"
s.split("char")	s="1,2,3,4,5"; x=s.split(",");	x=1,2,3,4,5
s.toUpperCase()	s="abcde"; x=s.toUpperCase();	x="ABCDE"
s.toLowerCase()	s="ABCDE"; x=s.toLowerCase();	x="abcde"

- [Ex] java1-1-string.htm

Variables

- **Global & local variables**

```
<script language="JavaScript">
    var x, y, z;
    ……
    function search(){
        var keyword, index;
        ……
    }
</script>
```

Math Object

- **Math object** is modeled on Java's Math class.
- It contains
  - **`abs`** absolute value
  - **`floor`** integer
  - **`log`** logarithm
  - **`max`** maximum
  - **`pow`** raise to a power
  - **`random`** random number
  - **`round`** nearest integer
  - **`sin`** sine
  - **`sqrt`** square root
- [Ex] java2-2-random-chang.htm

Statements

- *expression*
- **`if`**
- **`switch`**
- **`while`**
- **`do`**
- **`for`**
- **`break`**
- **`continue`**
- **`return`**

Var statement

- Defines variables within a function.
- Types are not specified.
- Initial values are optional.
  - `var name;`
  - `var nrErrors = 0;`
  - `var x="Hello Mr.";`

For statement

- Iterate through all of the elements of an array:

```
for (var i = 0; i < array.length; i += 1) {
    // within the loop,
    // i is the index of the current member
    // array[i] is the current element
}
```

For statement

- Iterate through all of the members of an object:

```
for (var name in object) {
    if (object.hasOwnProperty(name)) {
        // within the loop,
        // name is the key of current member
        // object[name] is the current value

    }
}
```

```
var  sum=0;
score= new Array(80,70, 60);
for (x in score){
   sum= sum + score[x];
} // final sum= 210
document.write("sum(80,70,60) = " +sum );
```

Switch statement

- Multiway branch
- The switch value does not need to be a number. It can be a string.
- The case values can be expressions.

Switch statement

```
switch (expression) {
case ';':
case ',':
case '.':
    punctuation();
    break;
default:
    noneOfTheAbove();
}
```

[Ex] java2-1-object.htm

Break statement

- Statements can have labels.
- Break statements can refer to those labels.

```
– loop: for (;;) {
    • …
        – if (...) {
            » break loop;
        – }
    • …
– }
```

Function statement

- **function** *name* (*parameters*) **{**

  statements ;

  }

  - [ex] function **Welcome()**{
    alert("歡迎光臨");
    }

Reserved Words

- abstract
  boolean **break** byte
  case catch char class const **continue**
  debugger **default delete do** double
  else enum export extends
  false final **finally** float **for function**
  goto
  if implements import **in instanceof** int
  interface
  long
  native **new null**
  package private protected public
  return
  short static super **switch** synchronized
  this throw throws transient **true try typeof**
  var volatile **void**
  while with

Arrays

- **Examples:**
  - `var names= new Array("國王","大饅頭","賓果");`
  - `var visit= Array(110);`
- **Array** inherits from **Object**.
- Indexes are converted to strings and used as names for retrieving values.
- Very efficient for sparse arrays.
- Not very efficient in most other cases.
- One advantage: No need to provide a length or type when creating an array.

`length`

- Arrays, unlike objects, have a special **`length`** member.
- It is always 1 larger than the highest integer subscript.
- It allows use of the traditional **`for`** statement.
  - **`for (i = 0; i < a.length; i += 1) {`**
    - …
  - **`}`**

Array Literals

- An array literal uses **`[]`**
- It can contain any number of expressions, separated by commas
  - **`myList = ['oats', 'peas', 'beans'];`**
- New items can be appended
  - **`myList[myList.length] = 'barley';`**
- The dot notation should not be used with arrays.
- **`[]`** is preferred to **`new Array()`**.

Array Methods

- [Ex] java3-1-array.htm
  - **concat**
  - **join**
  - **pop**
  - **push**
  - **slice(start,end)**
  - **sort**
  - **splice(index,n,add_element1，add_element2,…)**

Deleting Elements

```
delete array[number]
```

- Removes the element, but leaves a hole in the numbering.

```
array.splice(number, 1)
```

- Removes the element and renumbers all the following elements.

Deleting Elements

- `myArray = ['a', 'b', 'c', 'd'];`
- `delete myArray[1];`
  - `['a', undefined, 'c', 'd']`
- `myArray.splice(1, 1);`
  - `['a', 'c', 'd']`

Function

- Statement

```
function name (parameters) {
        statements ;
}
```

- Array Sorting & Searching
  - **Bubble Sort & Binary Search**
  - java3-2-array-sort-search.htm

```
<html>
<head>
<script language="javascript">
  function bubble_sort(list){
   ……
  }
  function bsearch(key,list){
   ……
  }
  </script>
</head>

<body>
<script language="javascript">
  var dlist= new Array("國王","大饅頭",
                       "瑪麗");
  bubble_sort(dlist);
  query= prompt("Type a dog name: ");
  bsearch(query,dlist);
</script>
</body>
</html>
```

Argument

```
function arg_sum(){
   var i, sum = 0;
   for (i=0;i < arguments.length; i++)
   { sum += arguments[i];
   }
   return sum;
}
```