Introduction to JavaScript – Part Two

Website Development 2

Lecture 1 Outline

- String Manipulation
- Math Object

String Manipulation

- Examples of strings are as follows:

```
var string1 = "blue";
var string2= " Today is Monday";
var string3 = "12";
```

String Manipulation

- String Manipulation allows us to:
  - Combine these strings into a sentence i.e. take these strings and concatenate them into one.
  - Break a string into smaller ones.
  - Convert a string into upper case or lowercase.
  - See if a particular character exists in a string.
  - Find the length of a string.
  - Convert a string into a number.

String Manipulation

- In addition to the concatenation operator (+) JavaScript supports several advanced string operations as well.

- These functions are accessed by referring to various methods of the String object.

- Moreover, this object also contains the ‘length’ property.

String Manipulation Example

```
name = "Bhola";
document.write(" The length of the string ‘name’ is ",
                                                        name.length ) ;
```

```
The length of the string ‘name’ is 5
```

String Methods

FORMAT: *string.methodName( )*

EXAMPLE:

```
name = "Bhola";
document.write(name);
document.write(name.toUpperCase()) ;
```

```
BholaBHOLA
```

String Methods: All Others

toLowerCase()

toUpperCase()

charAt(*n*)

substring(*n, m*)

indexOf(*substring, n*)

lastIndexOf(*substring, n*)

split(*delimiter*)

toLowerCase(), toUpperCase()

```
person = "Bhola" ;
document.write(person) ;
document.write(person.toLowerCase());
document.write(person.toUpperCase());
```

```
BholabholaBHOLA
```

charAt(*n*)

- Returns a string containing the character at position *n* (note that the position of the 1st character is 0).

```
mister = "Bhola" ;
document.write( mister.charAt(0));
document.write( mister.charAt(2));
```

```
Bo
```

substring(*n, m*)

- Returns a string containing characters copied from positions *n* to *m – 1.*

```
s = "Bhola" ;
document.write(s.substring(1, 3));
document.write(s.substring(0, s.length));
```

```
hoBhola
```

indexOf(*searchstring, n*)

- Returns the position of the first occurrence of *searchstring*. The search begins at character 0 unless specified by a value of N.
- -1 is returned if the *searchstring* is **not** found.

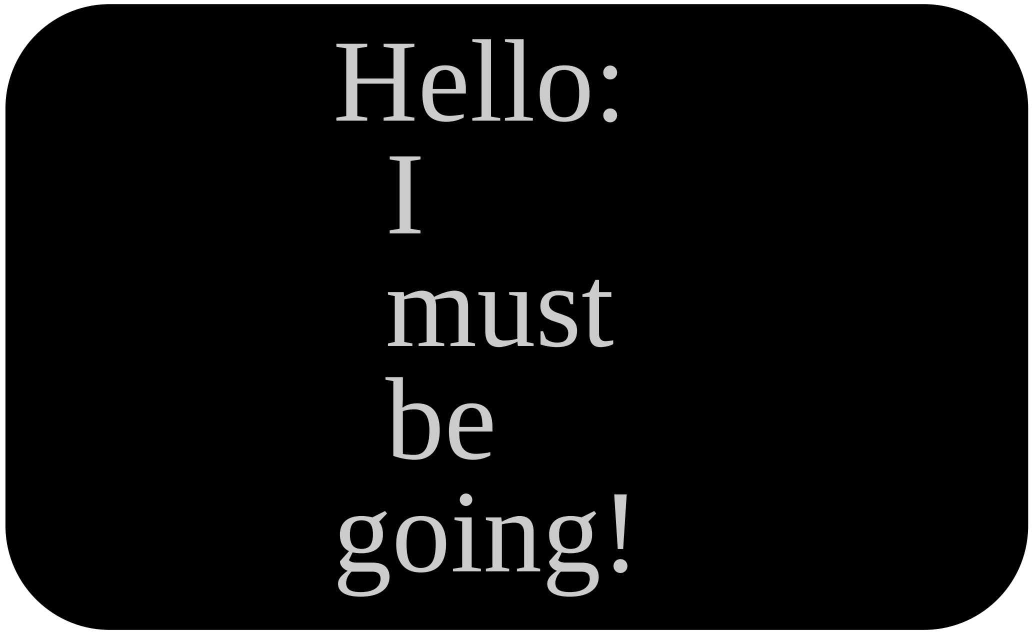
```
s = "Bhola" ;
document.write(s.indexOf("ola"));
document.write(s.indexOf("z"));
```

2-1

split(*delimiter*)

- Returns an array of strings, created by splitting string into substrings, at *delimiter* boundaries.

```
s = "Hello: I must be going!" ;
data = new Array() ;
data = s.split("  ") ;
document.write("<TABLE>") ;
for( i in data) {
     document.write("<TR><TD>", data[ i ], "</TD></TR>") ;
}
document.write("</TABLE>") ;
```



```
Hello:
I
must
be
going!
```

Automatic Conversion to Strings

- Whenever a non-string is used where JavaScript is expecting a string, it converts that non-string into a string.
- Example:
  - The document.write() method expects a string (or several strings, separated by commas) as its argument.
  - When a number or a Boolean is passed as an argument to this method, JavaScript automatically converts it into a string before writing it onto the document.

The ‘+’ Operator

- When ‘+’ is used with numeric operands, it adds them.
- When it is used with string operands, it concatenates them.
- When one operand is a string, and the other is not, the non-string will first be converted to a string and then the two strings will be concatenated.

The ‘+’ Operator: Examples

```
document.write(2 + 3) ;
```

```
5
```

```
document.write("2" + "3") ;
```

```
23
```

```
document.write("2" + 3) ;
```

```
23
```

Strings In Mathematical Expressions

- When a string is used in a mathematical context, if appropriate, JavaScript first converts it into a number. Otherwise, a “NaN” is the result.

```
document.write("2" * 3) ;
```

```
6
```

```
document.write("2" + 3) ;
```

```
23
```

The ‘toString’ Method

- The *toString()* method explicitly converts the input to a string.

- EXAMPLE:

  Convert 100.553478 into a currency format

```
a = 100.553478 ;
b = a.toString() ;
decimalPos = b.indexOf(".", 0) ;
c = b.substring(0, decimalPos + 3) ;
document.write(c) ;
```

```
100.55
```

Lecture 1 Outline

- String Manipulation
- Math Object

JavaScript Math Object

- In addition to the simple arithmetic operations (e.g. +, *, etc) JavaScript supports several advanced mathematical operations as well.

- These functions are accessed by referring to various methods of the **Math** object.

- Moreover, this object also contains several useful mathematical constants as its properties. For example Math.PI.

Methods

sin(r)
cos(r)
tan(r)
asin(x)
acos(x)
atan(x)
atan2(x, y)

sqrt(x)
pow(x, y)

exp(x)
log(x)

round(x)
floor(x)
ceil(x)

abs(x)

max(x, y)
min(x, y)

random()

sqrt(x)	pow(x, y)
Returns the square root of x	Returns x raised to the power y
Math.sqrt(9) → 3	Math.pow(2, 3) → 8

round(x)	floor(x)	ceil(x)
Returns integer nearest to x	Returns largest integer that is less than or equal to x	Returns smallest integer that is greater than or equal to x
1.1 → 1 12.5 → 13 12.9 → 13	1.1 → 1 12.5 → 12 12.9 → 12	1.1 → 2 12.5 → 13 12.9 → 13

abs(x)

Returns the absolute value of x

1.1 → 1.1

-12.5 → 12.5

0 → 0

min(x, y)	max(x, y)
Returns the smaller of x and y	Returns the larger of x and y
2, 4 → 2 -12, -5 → -12	2, 4 → 4 -12, -5 → -5

random()

Returns a randomly-selected, floating-point number between 0 and 1

Math.random( ) → 0.9601111965589273

random( ): Example

- Write JavaScript code that will display the result of the rolling of a 6-sided dice on user command.
- If you want to get a random number between 1 and another number, just multiply the random() method by the uppermost number and add 1 to the total.
- For Example: to generate a random number from 1 to 6:

```
var mynumber =Math.floor(Math.random()* 6 + 1);
```