SOEN 287

Chapter 4: JavaScript (1)

Dr. Yuhong Yan
CSE, Concordia University
Winter, 2023

Topics

- History
- Basic Syntax

This set of sides is modified from the slides accompanied the textbook.

History

- Originally developed by Branden Eich (Netscape), as LiveScript
- Became a joint venture of Netscape and Sun in 1995, renamed JavaScript
- Standardized by the European Computer Manufacturers Association as ECMA-262 (also ISO 16262)
  - Thus JavaScript also called ECMAScript (ES)
- The Original JavaScript ES1 ES2 ES3 (1997-1999)
- The First Main Revision ES5 (2009)
- The Second Revision ES6 (2015)
- All Yearly Additions (2016, 2017, 2018, 2019, 2020)

JavaScript vs Java

- JavaScript and Java are only related through syntax
  - JavaScript is dynamically typed
  - JavaScript's support for objects is very different
- Both are very powerful programming languages

JavaScript for Web Programming

- One of the three must learn languages
  - HTML5
  - CSS
  - JavaScript
- JavaScript for Web platform
  - Client side and server side
  - Speed
  - Gadgets
  - Mashups

JavaScript Features - Crockford

- Load and go delivery
  - case sensitive
- Loose typing (or dynamic typing)
- Objects as general containers
    - root object is `Object`
    - add properties to object, clone the objects
    - object are accessed through references
- Prototypal inheritance
  - no polymorphism
- Lambda
- Linkage though global variables

What tools you need to learn JavaScript

- Text editor
- Web browser
- No need Web server

General Syntax

- Import a JavaScript file

```
 <script type = "text/javaScript"
             src = "myScript.js">
</script>
```

- Embed JavaScript code

```
 <script type = "text/javaScript">
   <!–
   -- JavaScript script –
   //-->
</script>
```

- JavaScript comments: both `//` and `/* … */`

Hello World with JavaScript

```
<html lang = "en">
  <head>
    <title> Hello world </title>
    <meta charset = "utf-8" />
  </head>
  <body>
    <script>
      <!--
      document.write("Hello, fellow Web
programmers!");
      // -->
    </script>
  </body>
</html>
```

→**SHOW** `chap4/helloworld.html` **and display**

Now try to run JavaScript from your browser

Operations

- Numeric operators `++`, `--`, `+`, `-`, `*`, `/`, `%`
- The `Math` Object provides `floor`, `round`, `max`, `min`, trig functions, etc.
  - e.g., `Math.cos(x)`

→**SHOW `helloworld-variable.html` and display**

Primitives, Operations and Expressions

- The five primitive types: Number, String, Boolean, Undefined, or Null
  - Coerce
- The wrapper objects (`Number`, `String`, and `Boolean`)
- Example of numbers:

  72, 7.2, -72, 7E2, 7e2, 7.2E-2
- Example of strings:

  "Tuesday", 'Tuesday\n', 'Sam\'s work', "C:\\root"
- Boolean values are `true` and `false`
- The only Null value is `null`
- The only Undefined value is `undefined`

→**SHOW `hello-test.html` and display**

The `Number` Object

- `MAX_VALUE`, `MIN_VALUE`, `NaN`, `POSITIVE_INFINITY`, `NEGATIVE_INFINITY`, `PI`
- e.g., `Number.MAX_VALUE`
- An arithmetic operation that creates overflow returns `NaN`
- `NaN` is not == to any number, not even itself
- Test for it with `isNaN(x)`
- `Number` object has the method, `toString`

4 Ways to Declare a JavaScript Variable

- Using var
- Using let
  - Added to JavaScript in 2015
- Using const
  - Added to JavaScript in 2015
- Using nothing

String Operations

- Operator: +
- Coercion is used:

  "Jan " + 2010, 7* ‘3’

- Explicit conversions
  - Use the `String` and `Number` constructors
  - Use `toString` method of numbers
  - Use `parseInt` and `parseFloat` on string

```
var num = 6;
var str = String(num);
var str2 = num.toString();
var n1 = Number("6");
var n2 = parseInt("6");
```

→**SHOW** `hello-test2.html` **and display**

+: both plus and string concatenation -- pitfall

```
1 + 2
"1" + 2
1 + "2"
"1" + "2"
1+ " bird"
1+2+ "birds"
```

+: both plus and string concatenation

```
1 + 2
"1" + 2
1 + "2"
"1" + "2"
1+ " bird"
1+2+ "birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

+: both plus and string concatenation

```
1 + 2  = 3
"1" + 2
1 + "2"
"1" + "2"
1+ " bird"
1+2+ "birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

+: both plus and string concatenation

```
1 + 2  = 3
"1" + 2 = "12"
1 + "2"
"1" + "2"
1+ " bird"
1+2+ "birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

+: both plus and string concatenation

```
1 + 2  = 3
"1" + 2 = "12"
1 + "2" = "12"
"1" + "2"
1+ " bird"
1+2+ "birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

+: both plus and string concatenation

```
1 + 2  = 3
"1" + 2 = "12"
1 + "2" = "12"
"1" + "2" = "12"
1+ " bird"
1+2+ "birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

+: both plus and string concatenation

```
1 + 2  = 3
"1" + 2 = "12"
1 + "2" = "12"
"1" + "2" = "12"
1+ " bird" = "1 bird"
1+2+ "birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

+: both plus and string concatenation

```
1 + 2  = 3
"1" + 2 = "12"
1 + "2" = "12"
"1" + "2" = "12"
1+ " bird" = "1 bird"
1+2+ "birds" = "3birds"
```

- Only both operands are numbers, + is addition, otherwise string concatenation.

iClicker question

- What is the result of `'$' + 3 + 4` ?
  A. $7
  B. $34
  C. error
  D. undefined

Answer: B

Other operators in this case?

```
11 < 2
"11" < 2
11 < "2"
"11" < "2"
11 < "bird"
11 < 2+ "birds"
```

Other operators in this case?

```
11 < 2
"11" < 2
11 < "2"
"11" < "2"
11 < "bird"
11 < 2+ "birds"
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

Other operators in this case?

```
11 < 2                      false
"11" < 2
11 < "2"
"11" < "2"
11 < "bird"
11 < 2+ "birds"
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

Other operators in this case?

```
11 < 2                false
"11" < 2              false
11 < "2"
"11" < "2"
11 < "bird"
11 < 2+ "birds"
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

Other operators in this case?

```
11 < 2                    false
"11" < 2                  false
11 < "2"                  false
"11" < "2"
11 < "bird"
11 < 2+ "birds"
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

Other operators in this case?

```
11 < 2                  false
"11" < 2                false
11 < "2"                false
"11" < "2"              true
11 < "bird"
11 < 2+ "birds"
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

Other operators in this case?

```
11 < 2                    false
"11" < 2                  false
11 < "2"                  false
"11" < "2"                true
11 < "bird"               false
11 < 2+ "birds"
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

Other operators in this case?

```
11 < 2                  false
"11" < 2                false
11 < "2"                false
"11" < "2"              true
11 < "bird"             false
11 < 2+ "birds"         false
```

- If one operand is number, and the other can be converted to a number, < is a number comparison,
- If one operand is number, and the other cannot converted to a number, false all the time.
- If two operands are string, < is a string comparison

iClicker question

- What is the result of `'$' + 3 < 4` ?

  A. false

  B. true

  C. error

  D. undefined

Answer: A

Other operators in this case?

```
11 * 2
"11" * 2
11 * "2"
"11" * "2"
11 * "2bird"
```

Other operators in this case?

```
11 * 2
"11" * 2
11 * "2"
"11" * "2"
11 * "2bird"
```

- If operands are numbers, or all can be converted to a number, * is a number multiply
- If one operand is number, and the other cannot converted to a number, NaN all the time.

Other operators in this case?

```
11 * 2                  22
"11" * 2
11 * "2"
"11" * "2"
11 * "2bird"
```

- If operands are numbers, or all can be converted to a number, * is a number multiply
- If one operand is number, and the other cannot converted to a number, NaN all the time.

Other operators in this case?

```
11 * 2                  22
"11" * 2                22
11 * "2"
"11" * "2"
11 * "2bird"
```

- If operands are numbers, or all can be converted to a number, * is a number multiply
- If one operand is number, and the other cannot converted to a number, NaN all the time.

Other operators in this case?

```
11 * 2                22
"11" * 2              22
11 * "2"              22
"11" * "2"
11 * "2bird"
```

- If operands are numbers, or all can be converted to a number, * is a number multiply
- If one operand is number, and the other cannot converted to a number, NaN all the time.

Other operators in this case?

```
11 * 2                22
"11" * 2              22
11 * "2"              22
"11" * "2"            22
11 * "2bird"
```

- If operands are numbers, or all can be converted to a number, * is a number multiply
- If one operand is number, and the other cannot converted to a number, NaN all the time.

Other operators in this case?

```
11 * 2                  22
"11" * 2                22
11 * "2"                22
"11" * "2"              22
11 * "2bird"            NaN
```

- If operands are numbers, or all can be converted to a number, * is a number multiply
- If one operand is number, and the other cannot converted to a number, NaN all the time.

iClicker question

- What is the result of `'$'*3 < 4` ?
  A. false
  B. true
  C. error
  D. undefined

Answer: A

The End