Javascript, The Swiss Army Knife of Programming Languages

David Morcillo

23-11-2013

Stage 0: About me

- twitter.com/ultrayoshi
- github.com/ultrayoshi

Stage 1: Introduction

Features:

- Loosely typed language
- Object literal notation
- Prototypal inheritance
- Global variables
- Functions are first class objects

ECMAScript

The standard that defines JavaScript is the third edition of *ECMAScript Programming Language*.

Hello World

index.html

```
<html>
  <head>
    <script>
      document.writeln(‘‘Hello, world!’’);
    </script>
  </head>
  <body>
  </body>
</html>
```

Comments

Block comments formed with /* */ and line-ending comments starting with //. Example:

```
/*
  We are learning Javascript and comments are very important
*/
document.writeln(‘‘Hello World!’’); // Output: Hello World!
```

Names

Starts with a letter or underscore and optionally followed by on or more letters, digits or underscores. Beware of some reserved words.

```
bullet           // valid     _mana          // valid
weapon           // valid     life_          // valid
3force           // invalid   lucky42        // valid
rocket-launcher  // invalid   Hammer         // valid
grenade_launcher // valid     hammer         // valid, case sensitive
```

Numbers

Single number type represented internally as 64-bit floating point.

```
1
1.0
3.141516
10e5
-5E-10
.123456
1/0 // Output: Infinity
0/0 // Output: NaN
```

Strings

Can be wrapped in single quotes or double quotes. It can contains 0 or more characters. All characters in Javascript are 16 bits wide.

```
‘‘Hello World’’
’Hello World’
‘‘This is\n a multiline string’’
’You can write ‘‘ on single quotes string’
```

Variables

Use the `var` keyword followed by a name to declare a variable. When used inside of a function, the `var` statement defines the function's private variables.

```
var player; // variable player declared on a global scope

function test() {
    /*
        Variable enemy is visible on this function but
        we have access to variable player.
    */
    var enemy;

    function test2() {
        /*
            Variable bullet is visible on this function but
            we have access to player and enemy variables.
        */
        var bullet;
    }
}
```

if, else

```
var testOk = true;

if (testOk) {
    console.log(‘‘Captain obvious’’);
} else {
    console.log(‘‘I’m bored’’);
}
```

Here are the *falsy* values:

- `false`
- `null`
- `undefined`
- The empty string
- The number 0
- The number NaN

All other values are *truthy*.

switch

```
var weapon = ‘‘rocketlauncher’’;

switch(weapon) {
    case ‘‘pistol’’:
        console.log(‘‘piu piu’’);
        break;
    case ‘‘shotgun’’:
        console.log(‘‘paaam!’’);
        break;
    case ‘‘rocketlauncher’’
        console.log(‘‘BOOOOM!’’);
        break;
    default:
        console.log(‘‘falcon punch!’’);
        break;
}
```

while, do while

```
var counter = 0;

while (counter < 10) { // Ends when counter is equal to 10
    console.log(counter);
    counter += 1;
}

do {
    console.log(counter);
    i -= 1;
} while(counter > 0); // Ends when counter is equal to 0
```

for

```
var i;

for (i = 0; i < 10; i += 1)
    console.log(i);
}
```

Literals

```
var score = 9000;                                   // Number
var name = ‘‘David’’;                               // String
var weapons = [‘‘Pistol’’, ‘‘Shotgun’’, ‘‘Sword’’]; // Array
var player = {                                      // Object
    name: ‘‘Megaman’’,
    lifes: 3,
    weapons: [‘‘Buster’’, ‘‘Laser’’]
};
var run = function (param1, param2) {               // Function
};
var validation = /^pro/;                            // Regexp
```

Objects

- Objects in Javascript are mutable keyed collections.
- Arrays, functions and regular expressions are objects.
- A property name can be any string.
- Objects can inherit properties of another through its prototype.

Prototype

Every object is linked to a prototype object form which it can inherit properties. All objects created from object literals are linked to `Object.prototype`.
The prototype link is used only in retrieval. If we try to retrieve a property value from an object, and if the object lacks the property name, then Javascript attempts to retrieve the property value from the prototype object.