JS : Expressions et opérateurs

L2 MPCIE - UE Développement Web

David Lesaint

david.lesaint@univ-angers.fr

Janvier 2019

Les opérateurs

JS utilise des opérateurs unaires, binaires et l'opérateur ternaire conditionnel

Affectation	`=`	`+=`	`-=.`	`*=`	`/=`	`%=`
	`**=`	`<<=`	`>>=`	`>>>=`	`&=`	`^=`
	`|=`					
Comparaison	`==`	`===`	`!=`	`!==`	`>`	`<`
	`>=`	`<=`				
Arithmétique	`+`	`-`	`*`	`/`	`%`	`++`
	`--`	`-`	`**`			
Binaires	`&`	`|`	`~`	`^`	`<<`	`>>`
	`>>>`					
Logiques	`&&`	`||`	`!`			
Chaînes	`+`	`+=`				
Ternaire	`?:`					
Virgule	`,`					
Unaires	`delete`	`typeof`	`void`			
Relationnels	`in`	`instanceof`				

Affectation par décomposition

operateurs-affectation-decomposition.js

```
1 [a, b] = [10, 20];
2 console.log(a);
3 // expected output: 10
4 console.log(b);
5 // expected output: 20
6
7 [a, b, ...rest] = [10, 20, 30, 40, 50];
8 console.log(rest);
9 // expected output: [30,40,50]
```

Les opérateurs de comparaison

operateurs-comparaison.js

```
1  //Opérateurs de comparaison
2  console.log(1 === 2 - 1);// true
3  console.log(1 !== 2);// true
4
5  console.log(1 !== '1');// true
6  console.log(1 == '1');// true : transtypage
7  console.log(1 == '1.0');// true : transtypage
8  console.log(1 != '1.01');// true : transtypage
9  console.log(1 != '1 A');// true : transtypage
10 console.log(0 == '');// true : transtypage
11
12 console.log('ab' === 'a' + 'b');// true
13
14 console.log(undefined !== null);// true
15 console.log(undefined == null);// true : transtypage
16
17 console.log('0' < 'A');// true : comp. lexico. basée sur Unicode
18 console.log('Z' < 'a');// true
19
20 console.log('ab' < 'ac');// true
21 console.log('ab' < 'abc');// true
22
23 console.log(false < true);// true
```

Les opérateurs arithmétiques

operateurs-arithmetiques.js

```
1 // Opérateurs arithmétiques
2 var a = 5;
3 console.log(a / 2);// 2.5
4 console.log(Math.floor(a / 2));// 2
5 console.log(Math.ceil(a / 2));// 3
6 console.log(a % 2);// 3
7 console.log(++a);// 6
8 console.log(--a);// 5
```

Les opérateurs binaires

- Opèrent sur la représentation en 32 bits des nombres en arguments.
- Renvoient des valeurs numériques.

operateurs-binaires.js

```
1  // Opérateurs sur bitsets
2  var x = parseInt(    '0011', 2);
3  var y = parseInt('10000001', 2);
4  console.log(x);// 3
5  console.log(x.toString(2));// 11
6  console.log(y);// 129
7  console.log(y.toString(2));// 10000001
8  console.log(~x);// -4
9  console.log((~x).toString(2));// -100
10 console.log(x & y);// 1
11 console.log(x | y);// 131
12 console.log(x << 1);// 6
13 console.log(y >> 1);// 64
```

Les opérateurs logiques

Peuvent être appliqués à des valeurs non-booléennes

- `&&` renvoie son 1er opérande s'il peut être converti à `false`, son 2nd sinon.
- `||` renvoie son 1er opérande s'il peut être converti à `true`, son 2nd sinon.
- `!` renvoie `false` si son opérande peut être convertit à `true`, `true` sinon.

Les opérateurs logiques

operateurs-logiques.js

```
1  // Opérateurs logiques
2  console.log(!false);// true
3  console.log(!(!'faux'));// true
4  console.log(!!'faux');// true
5  console.log(!'');// true
6  console.log(!undefined);// true
7
8  console.log('I' || false);// 'I'
9  console.log(false || 'I');// 'I'
10 console.log('' || false);// false
11 console.log(false || '');//
12
13 var x = 2, y;
14 var z = y || x;
15 console.log(z);//2
16
17 console.log('a' && 'b'); //b
18 console.log('' && 'a'); //
19 console.log('a' && ''); //
```

Les opérateurs de chaînes et l'opérateur conditionnel

operateurs-conditionnel.js

```
1 // Les opérateurs de concaténation
2 console.log("ma " + "chaîne"); // ma chaîne
3 var maChaîne = "alpha";
4 maChaîne += "bet";
5 console.log(maChaîne);// alphabet
6
7 // L'opérateur ternaire conditionnel
8 var statut = (âge >= 18) ? "adulte" : "mineur";
```

L'opérateur virgule

- Evalue ses deux opérandes et renvoie la valeur du second.
- Utilisé pour séparer les incréments de compteurs dans les boucles.

operateurs-virgule.js

```
1 var x = [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ]
2 var a = [ x, x, x, x, x ];
3
4 for (var i = 0, j = 9; i <= j; i++, j--)
5   console.log("a[" + i + "][" + j + "]= " + a[i][j]);
```

Les opérateurs unaires

delete

Pour supprimer un objet, une propriété d'objet, ou un élément de tableau à partir de sa position (longueur du tableau non modifiée !)

operateurs-unaires.js

```
1  x = 42;
2  delete x; // true (suppression possible si déclaration implicite)
3  var y = 43;
4  delete y; // false (suppression impossible si déclaration avec var si ce n'est pas une propriété)
5  delete Math.PI; // false (suppression impossible pour les propriétés pré-définies)
6  monobj = new Number();
7  monobj.h = 4; // création de la propriété h
8  delete monobj.h; // true (suppression possible des propriétés définies par l'utilisateur)
9  delete monobj; // true (suppression possible si déclaration implicite)
10
11 var arbres = new Array("sequoia", "laurier", "cèdre", "chêne", "érable");
12 delete arbres[3];
13 if (3 in arbres) {
14   // Ceci ne sera pas exécuté
15 }
16
17 // Alternative
18 var trees = new Array("sequoia", "laurier", "cèdre", "chêne", "érable");
19 trees[3] = undefined;
20 if (3 in trees) {
21   // Ceci sera exécuté
22 }
```

Les opérateurs unaires

typeof

Pour déterminer le type d'une variable.

operateurs-unaires-typeof.js

```
1  var b = true;
2  console.log(typeof b); // boolean
3  var s = 'coucou';
4  console.log(typeof s); // string
5  var i = 10;
6  console.log(typeof i); // number
7  var k;
8  console.log(typeof k); // undefined
9  var tab = [ 1, 2 ];
10 console.log(typeof tab); // object
11 var p = null;
12 console.log(typeof p); // object
13 var reg = /\w+/;
14 console.log(typeof reg); // object
15 function f() {
16 }
17 console.log(typeof f); // function
```

Les opérateurs relationnels

in

Renvoie `true` si la propriété indiquée fait partie de l'objet donné.

operateurs-relationnels-in.js

```
1  // Tableaux
2  var arbres = new Array("sequoia", "laurier", "cèdre", "chêne", "érable");
3  0 in arbres; // true
4  3 in arbres; // true
5  6 in arbres; // false
6  "laurier" in arbres; // false (l'opérateur se base sur l'indice et pas sur la valeur)
7  "length" in arbres; // true (length est une propriété d'un objet Array)
8
9  // Objets pré-définis
10 "PI" in Math; // true
11 var myString = new String("coral");
12 "length" in myString; // true
13
14 // Objets définis par l'utilisateur
15 var maVoiture = {
16   fabricant : "Honda",
17   modèle : "Accord",
18   year : 1998
19 };
20 "fabricant" in maVoiture; // true
21 "modèle" in maVoiture; // true
```

Les opérateurs relationnels

`instanceof`

Renvoie `true` si l'objet donné est du type indiqué.

operateurs-relationnels-instanceof.js

```
1 var jour = new Date(2007, 01, 22);
2 if (jour instanceof Date) {
3    // instructions à exécuter
4 }
```