T5 : La conversion entre types de valeur, comment fait-on ? Les particularités…

Les différents types

En JavaScript, il y a 5 différents types de données qui peuvent contenir des valeurs:

- String
- Number
- Boolean
- Object
- Function

Il existe différents types d'objets comme **Object**, **Date** et **Array**, mais il en existe d'autres.
Et les types de données 2 qui ne peuvent pas contenir des valeurs: **Null** et **Undefined**

Les différentes fonctions

1. Typeof

   Elle permet de **trouver le type** de notre exemple. Si le type de la donnée est un tableau ou une date, alors **typeof** retournera forcément un objet et ne pourra définir le tableau ou la date.

```
typeof "John"                 // Returns string
typeof 3.14                   // Returns number
typeof NaN                    // Returns number
typeof false                  // Returns boolean
typeof [1,2,3,4]              // Returns object
typeof {name:'John', age:34}  // Returns object
typeof new Date()             // Returns object
typeof function () {}         // Returns function
typeof myCar                  // Returns undefined (if myCar is not declared)
typeof null                   // Returns object
```

(NaN : Not a Number)

Le type de données NaN est le number
Le type d'un de données array est un objet
Le type de données date est objet
Le type de données null est l' objet
Le type d'un de données undefined variable est undefined

2. String() et toString

   La méthode **String()** peut **convertir des nombres en chaînes**. Et la méthode du nombre **toString()** fait la même chose.

```
String(x)         // returns a string from a number variable x
String(123)       // returns a string from a number literal 123
String(100 + 23)  // returns a string from a number from an expression
```

```
x.toString()            // 123
(123).toString()        // 123
(100 + 23).toString()   // 123
```

Nous pouvons également convertir une date en chaîne.

```
Date().toString()   // returns Thu Jul 17 2014 15:38:19 GMT+0200 (W. Europe Daylight Time)
```

3. Number()

La méthode **Number()** peut **convertir des chaînes en nombres**.

```
Number("3.14")    // returns 3.14
Number(" ")       // returns 0
Number("")        // returns 0
Number("99 88")   // returns NaN
```

```
console.log(Number(undefined));    NaN
console.log(Number(null));    0
console.log(Number(true));    1
console.log(Number(false));    0
```

«3. 14 » se transforme en 3.14,
Les chaînes vide se convertissent à 0
Les autres chaines se transforme en NaN (qui n'est pas un nombre)

4. ParseInt et ParseFloat

ParseInt(string, base) comprend 2 arguments, « **string** » est **la chaine de caractère à convertir** et « **base** » est la **valeur de la base de numération**.
8 pour l'octal, 10 pour décimal et 16 pour l'hexadécimal.

```
alert(parseInt('10',10));          // Affiche 10

alert(typeof parseInt('10',10));   // Affiche number

alert(parseInt('dix',10));         // Affiche NaN

alert(typeof parseInt('dix',10)); // Affiche number

alert(parseInt('10.4',10));        // Affiche 10

alert(parseInt ('0xFa',16)) ;      // Affiche 250

alert(parseInt ('0456',8)) ;       // Affiche 302
```

La méthode **retourne un entier de type number** ou un code NaN si string n'est pas interprétable comme un number

ParseFloat(string) comprend 1 argument, « **string** » est **la chaine de caractère à convertir**.

```
alert(parseFloat('10.23'));           // Affiche 10.23

alert(typeof parseFloat('10.23'));    // Affiche number

alert(parseFloat('-10.23e2'));        // Affiche -1023

alert(parseFloat('Infinity'));        // Affiche Infinity

alert(typeof parseFloat('Infinity')); // Affiche number

alert(parseFloat('dix'));             // Affiche NaN

alert(typeof parseFloat('dix'));      // Affiche number

alert(parseFloat('10'));              // Affiche 10
```

La méthode **retourne un nombre décimal de type number** ou un code NaN si string n'est pas interprétable comme un number

5. Boolean()

```
console.log(Boolean(undefined));    false
console.log(Boolean(null));    false
console.log(Boolean(true));    true
console.log(Boolean(false));    false
console.log(Boolean(undefined));    false
console.log(Boolean(""));    false
```

La **méthode permet de retourner une valeur booléenne**