Les variables

Déclaration de variable

Les variables

Déclaration de variable

Il existe trois manières de déclarer des variables en JavaScript :

- Variable globale modifiable « **var** »
- Variable local modifiable « **let** »
- Variable local immuable « **const** »

	Global	Portée limité à la fonction	Portée limité au block	Ré-assignable	Re-déclarable	Hoisting
var	✔	✔	✘	✔	✔	✔
let	✘	✔	✔	✔	✘	✘
const	✘	✔	✔	✘	✘	✘

Nommer une variable

Le nom d'une variable peut contenir les caractères suivants :

- des caractères alphanumériques
- des underscores
- des dollars

Le nom de la variable doit commencer par un dollar, un underscore ou une lettre.

Bonne pratique :
La convention de nommage pour les variables est le « camelCase ».

Le typage

Les variables

Le typage

La JavaScript est un langage à typage dynamique.

Le type d'une variable sera défini dynamique lors de l'exécution en fonction du type du contenu qu'elle stockera.

De plus, il est possible que le type d'une variable change durant l'exécution

Bonne pratique :
Ne pas utiliser une même variable pour stocker plusieurs types de données !

Les types de valeurs primitifs

En JavaScript se sont des « pseudo-objets » qui permettent de définir les différents types de donnée avec leurs propriétés et leurs méthodes.

- Number : Un nombre (entier ou réel)
- BigInt : Un grand nombre entier
- String : Une chaîne de caractères
- Boolean : Type dont les valeurs possible sont « true » ou « false »
- Undefined : Variable non initialisée
- Null : Variable initialisée sans aucune valeur
- Object : Element complexe possédant plusieurs valeurs

Aperçu de quelques types de données

- Un nombre entier

```
const numberInteger = 42;
```

- Un nombre réel

```
const numberReal = 3.14;
```

- Une chaîne de caractères

```
const firstname = 'Della';
```

- Un booléen

```
const isValid = true;
```

- Un tableau de nombre

```
const tabNumber = [13, 42, 99, 42];
```

- Un tableau de chaîne de caractères

```
const tabName = ['Riri', 'Fifi', 'Zaza'];
```

- Une valeur « null »

```
const example = null;
```

- Un objet

```
const person = {
  firstname : 'Balthazar',
  lastname : 'Picsou'
};
```

Obtenir le type d'une variable

L'opérateur « typeof » permet d'obtenir le type d'une variable.

Le type est renvoyé sous la forme d'une chaine de caractere.

```
const response = 42;
const responseType = typeof response;

console.log(responseType); // → 'number'
```

Remarque : L'utilisation de l'opérateur « typeof » sur une variable de type « Null » renvoie la valeur « 'object' » pour des raisons historiques.

Obtenir le type d'une variable - Les objets

L'opérateur « typeof » ne peut pas indiquer le prototype d'un objet (tableau, date, ...). Celui-ci renverra toujours le résultat « 'object' ».

Pour connaître le prototype d'une variable de type « Object », il est possible :

- D'obtenir le nom du prototype via le propriété « constructor »
- De tester l'appartenance à un prototype via l'opérateur « instanceof »

```
const now = new Date();

console.log(typeof now);           // → 'object'
console.log(now.constructor.name); // → 'Date'
console.log(now instanceof Date);  // → true
```

Les valeurs prédéfinis

Les variables

Undefined

La valeur « undefined » est une propriété globale qui représente la primitive undefined.

Cette valeur primitive est affectée automatiquement :

- aux variables qui viennent d'être déclarée sans avoir été initialisée.
- aux paramètres de fonction qui ne possède pas de valeur par défaut.

Null

La valeur « null » est un littéral qui représente l'absence intentionnel de valeur pour une variable objet. C'est une valeur primitive du langage JavaScript.

Contrairement à la valeur « undefined », ce n'est pas une propriété globale.

NaN

La valeur « NaN » (Not a Number) est une une propriété globale qui permet de représenter une valeur numérique invalide.

Particularité de cette propriété « NaN » :

- Elle est de type « 'number' »

```
console.log(typeof NaN);     // → 'number'
```

- Les opérateurs d'égalité ne peuvent pas être utiliser sur celle-ci.
  Pour tester cette valeur, il possible d'utiliser la méthode global « isNaN(...) »

```
console.log(NaN === NaN);    // → false
console.log(isNaN(NaN));     // → true
```

Infinity

La valeur « Infinity » est une une propriété globale qui permet de représenter l'infini.

Comportement de la valeur « Infinity » :

- Tout nombre multiplié par Infinity sera égal à Infinity
- Zéro multiplié par Infinity sera égal à NaN
- NaN multiplié par Infinity sera égal à NaN
- Tout nombre divisé par Infinity sera égal à zéro
- Infinity divisé par Infinity sera égal à NaN
- Infinity divisé par un nombre sera égal à Infinity