


A small yellow square containing the letters 'JS' in a bold, dark grey, sans-serif font.

Web Developer Scripts Clients

A large yellow square containing the letters 'JS' in a bold, dark grey, sans-serif font.

02 – Les variables



Variables – Identificateurs

Une variable est un nom qui permet d'accéder à la mémoire utilisée par l'application

Elle est introduite en JS par l'un des trois mots-clés suivants :

- `var` : seule déclaration possible pour les versions d'ES<6. Désormais à éviter sauf cas particuliers.
- `const` : à utiliser pour toutes les valeurs constantes (ES>=6)
- `let` : à utiliser pour toutes les autres valeurs (ES>=6)



Variables – Identificateurs

Pour définir le nom d'une variable, nous disposons des caractères suivants :

- [a-z]
- [A-Z]
- [0-9]
- **_** et **\$**



Variables – Identificateurs

Le nom d'une variable :

- Doit être explicite
- Ne pas être trop long
- Ne peut pas commencer par un chiffre
- Est sensible à la casse
- Peut comporter des caractères accentués, mais il est préférable de ne pas les utiliser



Variables – Identificateurs

S'il est composé de plusieurs mots, il est préférable de les séparer par une majuscule :

```
let var_de_test;  
let varDeTest; // Style à privilégier
```



Variables – Déclaration

En JavaScript, les variables sont typées automatiquement, c'est-à-dire que le type de la donnée est déterminée par l'interpréteur et pas par le développeur

Pour déclarer une variable, il suffit de faire précéder l'identificateur par le mot-clé `let` ou `const` (fonctionne également avec `var`):

```
let varTest1;  
const test2;  
let _t3;
```



Variables – `let` ou `const` ?

Si vous désirez déclarer une constante, vous utiliserez `const`. Attention, vous devez l'initialiser à la déclaration car ensuite, par définition, il n'est plus possible de la modifier :

```
const constante_1 = 26;
```

```
// génère une erreur (TypeError)  
constante_1 = 10;
```

```
// génère une erreur (SyntaxError)  
const constante_1;
```

Pour tous les autres cas, vous utiliserez `let`.



Variables – Déclaration

La déclaration multiple :

```
let varTest1, test2, _t3;
```

Nous pouvons déclarer plusieurs variables en une seule instruction en séparant celles-ci par des virgules.



Variables – Déclaration

Il est possible d'initialiser la variable dès sa déclaration :

```
let varTest1 = 10;  
const varConst1 = 12;
```

Si elle n'est pas initialisée, elle vaut `undefined` :

```
let varTest1 = 10;  
let test2;  
console.log(varTest1);  
console.log(test2);
```



Variables – Déclaration

Pour déterminer le type d'une variable, on peut utiliser l'opérateur `typeof` :

```
let varEntier = 1;  
let varReel = 2.0;  
let varBool = true;  
console.log(typeof varEntier);  
console.log(typeof varReel);  
console.log(typeof varBool);
```



Variables – Déclaration

Pour déterminer le type d'une variable, on peut utiliser l'opérateur `typeof` :

```
let varCaractere = 'a';  
let varString = 'ab';  
let varUndefined;  
console.log(typeof varCaractere);  
console.log(typeof varString);  
console.log(typeof varUndefined);
```



Variables – Déclaration

On voit dans la console que ces six variables sont de quatre types :

- `number` : représente un nombre quel que soit son format
- `boolean` : `true` ou `false`
- `string` : représente une chaîne de caractères. Celle-ci peut être placée entre simple ou double quote

```
let varTest1 = 'val1', varTest2 = "val2";
```

- `undefined` : représente une variable dont le type n'est pas défini (ou qui ne contient pas de valeur). Variable déclarée sans valeur



Variables – Déclaration

D'autres types de JS :

- `null` : valeur vide volontaire
- `object` : `{}`, tableaux, dates, etc.
- `bigint`, `symbol` (cas avancés)



Variables – Déclaration

Test du typage dynamique :

```
let varTest = 1;  
  
console.log(typeof varTest);  
  
varTest = 'test';  
  
console.log(typeof varTest);
```

varTest **passe du type** number **au type** string **comme attendu**



Opérateurs arithmétiques

En JavaScript (comme dans beaucoup d'autres langages de programmation), il existe 6 opérations arithmétiques :

- Addition : +
- Soustraction : -
- Multiplication : *
- Division : /
- Modulo : %
- Exponentiation : **



Opérateurs arithmétiques – Addition

```
let terme1 = 2;  
let terme2 = 9;  
  
let somme = terme1 + terme2;  
  
console.log(somme);
```




Opérateurs arithmétiques – Soustraction

```
let terme1 = 2;  
let terme2 = 9;  
  
let difference = terme1 - terme2;  
  
console.log(difference);
```



Opérateurs arithmétiques – Multiplication

```
let facteur1 = 2;  
let facteur2 = 9;  
  
let produit = facteur1 * facteur2;  
  
console.log(produit);
```



Opérateurs arithmétiques – Division

```
let dividende = 9;  
let diviseur = 2;  
  
let quotient = dividende / diviseur;  
  
console.log(quotient);
```



Opérateurs arithmétiques – Division

Que se passe-t-il si on divise par zéro ?

```
let dividende = 9;  
let diviseur = 0;  
  
let quotient = dividende / diviseur;  
  
console.log(quotient);
```

On obtient la valeur `Infinity`



Opérateurs arithmétiques – Modulo

Le modulo nous donne le reste de la division entière de deux nombres :

```
let dividende = 19;  
let diviseur = 4;  
  
let reste = dividende % diviseur;  
  
console.log(reste);
```



Opér. arithmétiques – Exponentiation

L'exponentiation nous donne le résultat de la puissance d'un nombre par un autre :

```
let nombre = 4;  
let exposant = 3;  
  
let puissance = nombre ** exposant;  
  
console.log(puissance);
```



Opérateurs arithmétiques – Raccourcis

Il est possible d'utiliser les raccourcis suivants :

- `+=`
- `-=`
- `*=`
- `/=`
- `%=`



Opérateurs arithmétiques – Raccourcis

Par exemple :

```
terme1 += terme2;
```

// est équivalent à

```
terme1 = terme1 + terme2;
```




Affichage

Pour juxtaposer des chaînes de caractères ou des chaînes et des nombres, on réalisera une concaténation.

En JS, l'opérateur de concaténation est le +

```
let varTest1 = 10;  
  
console.log("La variable de test = " + varTest1);
```



Affichage

- Depuis ES6, pour la concaténation, on utilisera les « template literals ».
- Celles-ci sont délimitées par des backticks (attention, il s'agit d'accents graves, pas des guillemets) : `texte de test`.
- Permettent d'insérer des variables et des expressions directement au sein des caractères `\${}`

```
let varTest1 = 10;
```

```
console.log(`La variable de test = ${varTest1}`);
```



Affichage

Pour afficher le nom de l'utilisateur, on peut utiliser une nouvelle boîte de dialogue : `prompt`

```
let nomUtilisateur = prompt('Quel est votre nom ?') ;  
  
console.log(`Bonjour ${nomUtilisateur}`);
```

Par défaut, `prompt` retourne une chaîne de caractères



Affichage

Exercice 2.1.1

Créez une petite calculatrice pour additionner deux nombres saisis par l'utilisateur



Affichage

Puisque, par défaut `prompt` retourne une chaîne de caractères, si nous voulons additionner deux nombres, nous devons convertir la chaîne de caractères en un nombre :

- `parseInt` : pour convertir en un entier
- `parseFloat` : pour convertir en un réel



Affichage

```
let nombre1 = parseInt(prompt('Premier nombre à  
additionner : '));  
let nombre2 = parseInt(prompt('Deuxième nombre à  
additionner : '));  
  
let somme = nombre1 + nombre2;  
  
console.log(`nombre1 + nombre2 = ${somme}`);
```



Bonnes pratiques

- Toujours utiliser `const` par défaut : passer à `let` uniquement si la valeur doit changer
- Éviter `var` : obsolète, peut causer des erreurs de portée
- Indenter avec 4 espaces
- Noms explicites en camelCase
- Commenter uniquement quand nécessaire
- Initialiser les variables si possible : éviter `undefined` involontaire
- Utiliser les template literals : ``Bonjour ${prenom}`` au lieu de `"Bonjour " + prenom`



Exercices

2.2.1 Calculez le périmètre et l'aire d'un carré l'utilisateur donnant la valeur du côté

2.2.2 Calculez le périmètre et l'aire d'un rectangle l'utilisateur donnant la valeur de la longueur et de la largeur

2.2.3 Écrire un script qui établit une facture comportant une seule ligne (un seul produit en un nombre quelconque d'exemplaires). L'utilisateur précisera le prix unitaire, le nombre d'exemplaires, et le taux de TVA (en %). Une remise de 10% est accordée systématiquement (sur le prix HTVA). Le script affichera le montant HTVA, le montant de la TVA et le prix TVAC.



Exercices

2.2.4 Écrire un script qui calcule le nombre de billets ou de pièces de 1, 2, 5, 10, 20, 50, 100, 200, 500 euros nécessaires pour former un montant donné. Il calculera le nombre minimal de coupures pour former le montant donné par l'utilisateur

2.2.5 Écrire un script qui convertit en secondes une durée donnée en heures, minutes et secondes

2.2.6 Écrire un script qui convertit en heures, minutes et secondes une durée donnée en secondes

2.2.7 Écrire un script qui calcule le temps écoulé entre deux instants donnés d'une même journée (les deux instants sont donnés en heures, minutes et secondes)



Exercices

2.2.8 Écrire un script qui réalise la permutation de deux variables numériques. Les valeurs des variables sont affichées avant et après permutation

2.2.9 Écrire un script qui réalise la permutation circulaire à gauche de 3 variables numériques. Les valeurs des données sont affichées avant et après permutation.

Permutation circulaire à gauche des variables A, B, C = $A \leftarrow B \leftarrow C \leftarrow A$

2.2.10 Écrire un script qui réalise la permutation circulaire à droite de 3 variables numériques. Les valeurs des variables sont affichées avant et après permutation

Permutation circulaire à droite des variables A, B, C = $A \rightarrow B \rightarrow C \rightarrow A$