

JAVASCRIPT : LES BASES DU LANGAGE

JavaScript est un langage de programmation du Web. Il a été créé en 1995. Avec les comportements dynamiques qu'il introduit, comme les boîtes de dialogue qui s'affichent à l'écran, les formulaires interactifs ou encore les changements d'image au survol de la souris, le succès de JavaScript est immédiat. Depuis, le JavaScript fait l'objet de nombreuses évolutions et la dernière version a été officialisée en 2016 et constitue la septième édition du standard ECMA.

Son objectif est de dynamiser les pages Web et de les rendre interactives. Exemple un slider, grâce à JavaScript, nous pouvons réaliser un défilement automatique des images. Avec JS, toute la page (ou le DOM) est manipulable et suite à des événements créés par l'utilisateur, il est possible de modifier le code HTML et CSS.

Tous les navigateurs sur le marché sont capables d'exécuter JavaScript mais attention, ils n'ont pas tous le même moteur. C'est pourquoi il est important de tester son programme sur les principaux navigateurs.

Pour le moment, vous avez appris à taper des lignes de HTML et de CSS, ce sont des langages de description, c'est-à-dire que vous décrivez, par le biais de ces langages, ce que vous voulez voir apparaître dans le navigateur... et le navigateur vous suit...

Le JavaScript est quand à lui un langage de programmation impératifs, c'est-à-dire qu'il donne des ordres au navigateur :

- Affiche ceci !
- Passe à l'image suivante
- Si l'utilisateur clique là, alors tu fais ça !!
-

Ces ordres vont constituer un programme qui est en fait une liste d'instructions qui vont être lues et interprétées par un navigateur.

Exemple d'un calcul de TVA :

- Je prends mon montant HT
- Je détermine ma TVA :
 - Si mon article est alimentaire, je mets ma TVA à 5.5
 - Si mon article est non alimentaire, je mets ma TVA à 20
- Je calcule mon montant de TVA
- J'ajoute ce montant au montant HT pour obtenir mon montant TTC

QU'EST-CE QU'UN LANGAGE DE PROGRAMMATION ?

Un langage de programmation est une langue qui permet à l'être humain de dialoguer avec la machine.

En effet un ordinateur ne parle qu'en binaire, c'est-à-dire qu'avec des suites de 0 et de 1 :

```
0011010101100110110010011
```

Aucun être humain n'est capable d'interpréter du binaire, enfin si, votre prénom au grand max qui représente déjà des dizaines de caractères.

C'est là qu'interviennent les langages de programmation, ils sont un pont entre le binaire et l'humain, ils traduisent en binaire les instructions transmises par l'humain.

Il en existe beaucoup, les langages dédiés au Web sont JavaScript et PHP et pour chaque spécialité, il y a des langages dédiés : Python, C, C++, Ruby, perl ...

QU'EST-CE QUE LE DEVELOPPEMENT WEB ?

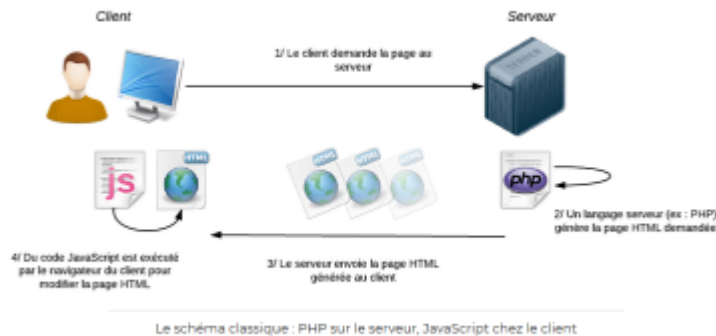
Le développement Web ne consiste pas seulement en la réalisation de sites Internet. Derrière de nombreux sites, il y a des applications, des back offices par exemple.

Et puis il y a tout le reste, des outils servant aux entreprises, qui n'ont pas forcément de vitrine. Cela peut être des gestionnaires de contrats, des formulaires d'inscriptions, de la comptabilité, des statistiques... Il est impossible de lister tout ce qui est possible de faire...

L'avantage de développer son application en format Web est sa portabilité, quoi de mieux qu'un outil qui est disponible partout dès lors que nous possédons une connexion Internet.

CLIENT / SERVEUR

Avant toute chose, il est important de comprendre la notion de client / serveur : Le client représente ici le navigateur :



OU ECRIRE DU JS ?

Deux possibilités :

- Dans le fichier html entre 2 balises `<script>.... </script>` (mauvaise pratique)
- Dans un fichier externe (à privilégier)

```

1  <!DOCTYPE html>
2  <html lang="fr">
3  <head>
4      <meta charset="utf-8">
5      <title>Mon premier fichier Javascript</title>
6  </head>
7  <body>
8      <h1>Bonjour</h1>
9      <script src="exemple1.js"></script>
10 </body>
11 </html>
    
```

Le lien vers ce fichier va être fait grâce à une balise script et un attribut src. Un fichier JavaScript est constitué d'un nom et d'une extension .js :

```
<script src='exemple.js'></script>
```

Ce lien peut être placé avant la fermeture de la balise head ou avant la fermeture de la balise body, tout dépend de ce qu'il contient.

LES COMMENTAIRES

Les commentaires sont très importants, ils ne vont pas être interprétés par le navigateur, nous pouvons donc y mettre ce que l'on veut. Ils servent à indiquer des informations sur l'utilité d'une instruction ou d'une fonction. L'objectif étant de s'y retrouver facilement lorsque l'on reprend en main un programme :

```
//Mon premier commentaire sur une seule ligne  
  
/* Mon commentaire  
sur plusieurs  
lignes..  
*/
```

AFFICHER DES INFORMATIONS

EN JS GRACE A `DOCUMENT.WRITE`

```
document.write("Bonjour");
```

Affiche une information directement sur notre page, dans le navigateur à l'endroit où le lien vers le fichier script est fait.

ALERT

```
alert("Bonjour");
```

Ouvre une boîte de dialogue (popup) pour afficher une info, très peu utilisée mais à connaître.

CONSOLE.LOG

```
console.log("Bonjour");
```

Affiche une info dans la console (F12), très utilisée lors d'un développement, elle affiche les erreurs. Ici, cela affichera simplement « Bonjour », mais parfois on peut s'en servir pour afficher le contenu d'une variable.

La console.log est très utilisé pour déboguer les programmes. Dès que votre programme ne fonctionne pas, pensez à appuyer sur F12 pour ouvrir la console, vous y trouverez peut être un indice sur l'erreur commise !

GESTION DES DONNEES

LES VARIABLES

Une variable est une sorte de petite boîte qui porte un nom et qui va contenir une valeur.

Elle doit être **déclarée** une et une seule fois dans un programme. Il existe 2 types de variables :

- Les **variables globales** : qui vont être utilisées dans tout le programme, elles doivent être déclarées à la base du code, en dehors de toute fonction.
- Les **variables locales** : qui ne servent que pour un programme ou une fonction en particulier. Elles doivent être déclarées au sein de ce programme ou de cette fonction.

Syntaxe pour déclarer une variable :

```
let maPremiereVariable;
```

Par convention, on utilise une règle de nommage pour nos variables, qui est également valable pour les fonctions. Il s'agit du **CamelCase**, en les liant sans espace ni ponctuation, et en mettant en capitale la première lettre de chaque mot sauf le premier. De manière générale, choisissez des noms qui veulent dire quelque chose. Evitons « variable1 », « variable2 »...

Attention **JS est sensible à la casse**, *mavariabLe* n'est pas la même chose que *maVariable*. JS ne l'interprétera pas comme étant la même variable.

Reprenons l'exemple précédent où nous affichions « Bonjour ». Nous avons mis cette chaîne de caractère directement dans le `console.log`, nous aurions pu aussi la stocker dans une variable afin de la réutiliser :

```
let maPremiereVariable = "Bonjour";
console.log(maPremiereVariable);
document.write(maPremiereVariable);
alert(maPremiereVariable);
```

Une variable, comme son nom l'indique, peut également changer de valeur à tout moment :

```
let maPremiereVariable = "Bonjour";
console.log(maPremiereVariable);
document.write(maPremiereVariable);

maPremiereVariable = "Salut";
alert(maPremiereVariable);
```

LES CONSTANTES

Une constante fonctionne comme une variable sauf qu'il faut lui attribuer une valeur à la déclaration et qu'ensuite on ne peut plus modifier cette valeur.

L'intérêt est de rendre plus clair le code. En lisant un projet, le développeur peut se demander pourquoi telle ou telle valeur obscure. En remplaçant cette valeur par une constante avec un nom précis, on documente le code.

Déclaration d'une constante :

```
const x = 5;
```

Essayons d'en modifier la valeur :

```
const x = 5;
x = 7;
```

Et voilà ce que l'on obtient en console :

```
► TypeError: invalid assignment to const `x` [En savoir plus]
```

LES TYPES

Quelles valeurs peuvent prendre mes variables :

- **String** ou chaîne de caractères: un texte qu'il faut mettre entre guillemets simples, doubles ou quotes.

Simple ou double, peu importe, cela revient à la même chose, par contre les quotes permettent plus de choses et notamment le retour à la ligne :

```
let text = "Coucou";
let otherText = 'Beuh';
let encoreOtherText = `
<p>Bonjour, ça va ?</p>
<p>Je m'appelle Cécile</p>
`;
```

- **Number** : peut contenir des entiers ou des réels, les virgules sont exprimées grâce à un point. Exemple 3,14 s'écrit 3.14. Sur des variables, il est possible de réaliser des calculs : + - * / %. Attention, ici pas de guillemets sinon le nombre sera considéré comme une chaîne de caractères
- **Booléen** : 2 valeurs sont possibles : true et false
- **Tableau** : appelé Array en JS, il contient plusieurs valeurs classées dans un tableau
- **Objet** : Il contient des propriétés et des méthodes

CONCATENATION.

La concaténation consiste en l'assemblage de texte et de variables. On utilise le symbole + pour concaténer. Le « + » étant également utilisé pour additionner, JS fait la différence en fonction du type de valeurs qui entourent le +. S'il s'agit de nombres, il va les additionner mais s'il s'agit de textes, il va les concaténer.

Exemple :

```
2 let nom = "CHALOUR";  
3 let prenom = "Leyla";  
4 console.log(nom + prenom);  
5
```

Donnera

CHAKOURLeyla

```
1  
2 let nom = "CHALOUR |";  
3 let prenom = "Leyla";  
4 console.log(nom + prenom);  
5
```

Et là c'est parfait !

CHAKOUR Leyla

CALCULS SUR DES VARIABLES

Il est possible de faire des calculs sur des variables de type Nombre :

- L'addition : +
- La soustraction : -

- La multiplication : *
- La division : /
- Le modulo : % (reste d'une division)
- Incrémenter une variable de 1 : ++
- Décrémenter une variable de 1 : --
- Incrémenter une variable de la valeur d'une seconde variable : +=
- Décrémenter une variable de la valeur d'une seconde variable : -=

Exemples:

```
let x = 6;
let y = 3;

let addition = x + y;
console.log(addition); //affiche 9

let soustraction = x - y;
console.log(soustraction); //affiche 3

let multiplication = x * y;
console.log(multiplication); //affiche 18

let division = x / y;
console.log(division); //affiche 2

let modulo = x % y;
console.log(modulo); //affiche 0
```

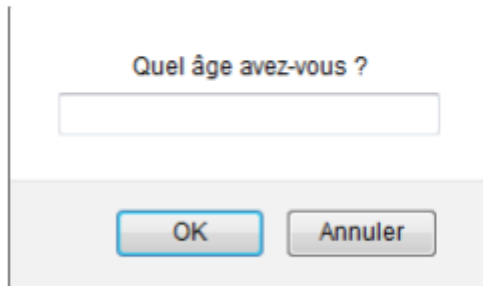
PROMPT

L'emploi de la fonction prompt va permettre de demander à l'utilisateur une information. Elle va ouvrir une boîte de dialogue qui contiendra une question et un champ vide pour que l'utilisateur puisse répondre.

Syntaxe :

```
let age = window.prompt("Quel âge avez-vous ?");  
console.log(age);
```

Affichera ceci :



Pour stocker la valeur saisie par l'utilisateur dans une variable, il faut attribuer ce prompt à une variable.

Ainsi, la valeur saisie par l'utilisateur sera enregistrée dans la variable « age ».

Attention, le prompt renvoie toujours une chaîne de caractères, ce qui signifie que si vous souhaitez faire des calculs avec cet âge, par exemple indiquer l'âge qu'aura la personne dans 1 an ($\text{age} + 1$), il va alors falloir modifier le type de la variable obtenue avec le prompt.

CONVERSION

Les fonctions `parseInt` et `parseFloat` servent à convertir une chaîne de caractères en nombre entier pour le premier et en nombre à virgules pour le second.

Exemple d'utilisation, lorsque l'on demande une information à l'utilisateur via un prompt, la valeur enregistrée est de type Chaîne de caractères. On ne peut donc pas faire de calculs dessus, il va falloir d'abord dire au programme qu'il s'agit d'un nombre :

```
age = parseInt(age);  
age ++;  
console.log(`L'année prochaine, vous aurez ${age} ans !`);
```