Résumé JavaScript

qkzk

JavaScript :

Quelques particularités de JavaScript :

- exécuté dans le navigateur, côté client. Par opposition aux calculs réalisés côté serveur (avec Python, PhP, node.js)
- javascript est un langage "orienté objet" comme presque tous les langages "majeurs" (à l'exception notable du langage C, le "parent" des langages modernes.)
- javascript permet, depuis peu, d'être exécuté côté serveur grâce à Node.js
- javascript réagit à des "événements". Un *événement* est une action qui se produit dans le système et auquel javascript peut répondre :

  `onmouseover` == "quand la souris passe sur cet élément..."

  `bouton.onmouseover = function(){ fais ceci }` : quand la souris passe au dessus du bouton, fais ceci.

Intégrer à une page html.

Il existe plusieurs moyens, le plus courant (et le meilleur) consiste à appeler un fichier `.js` depuis une page avec la balise `script`

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <script src="monscript.js"></script>
  </head>
  <body>
    <h1>ma page</h1>
  </body>
</html>
```

Et dans `monscript.js` se trouve le code JavaScript associé.

Syntaxe

Elle est dérivée du langage C (comme Java, C++, C# et tant d'autres). C'est une syntaxe que vous apprendrez forcement si vous décidez d'étudier l'informatique.

L'indentation est *optionnelle* mais indispensable si vous voulez comprendre ce que vous faîtes.

Commentaires

```
// commentaire sur une seule ligne. Ils commencent par //

/*
  Ou comme en CSS, sur plusieurs lignes
*/
```

Variables

Plusieurs manières de les déclarer :

```
a = 2; // variable globale
var b = 3; // variable locale dans une fonction (n'existera pas en dehors)
let c = 4; // variable n'existant que dans le bloc courant
```

Selectionner un élément d'une page

```
var elem = document.querySelector("#monid"); //
```

La variable `elem` est maintenant une référence vers l'élément ayant l'id `#monid` et toutes ses propriétés.

Modifer un contenu html

```
document.querySelector("#monid").innerHTML = "Thomas super fort";
```

On écrase le contenu HTML de l'élément `#monid` par ce qui se trouve à droite du signe `=`

Événéments

Un exemple :

Considérons le code html suivant :

```
<button>Mettre en rouge</button>
```

Et le javascript associé :

```
var btn = document.querySelector('button');

btn.onclick = function() {
  document.body.style.backgroundColor = '#FF00FF';
}
```

En cliquant sur bouton, il va devenir magenta :

- `#` : couleur en hexa,
- `FF` : rouge à fond.
- `00` : pas de vert,
- `FF` : bleu à fond.

Résultat magenta

Les fonctions

Au passage, on a rencontré ici une *fonction*.

On peut nommer les fonctions comme en Python :

```
function random(number) {
  let nombre = Math.floor(Math.random() * (number + 1));
  return nombre;
}
```

En détail :

- la fonction s'appelle `random`
- Elle prend un paramètre appelé `number`
- Elle renvoie le résultat d'un calcul contenu dans la variable `nombre`
- Le calcul consiste à :
  1. Tirer un nombre réel au hasard entre 0 et 1 (un flottant) avec Math.random()
  2. Le multiplier par (number + 1) (il est entre 0 et number + 1)
  3. L'arrondir à l'entier inférieur avec Math.floor(...)

On l'appelle avec :

```
var nb = random(15) // un entier aléatoire entre 0 et 15 inclus
```

Un analogue en Python serait :

```
from random import randint
nb = randint(0, 15)
```

Et oui, Python contient déjà cette fonction, inutile de la développer !

Il existe de nombreuses manières de déclarer des fonctions en JS.

Tests

```
if (condition) {
  executé si condition est vrai
}
else {
  executé si condition est faux
}
```

Les conditions entre `()`, les blocs exécutés entre `{ }`

Boucle While

```
while (condition){
  faire ceci
}
```

Par exemple :

```
var a = 2;
while (a < 10){
  faire des trucs;
  a = a + 1;
}
```

On rencontre souvent `a++` à la place de `a = a + 1`

C'est la boucle qui ressemble le plus à celle de Python

Boucle For

Il existe des dizaines de manières d'écrire une boucle for en JavaScript.

La plus courante est

```
for (let i = 0; i<5; i++){
  faire des trucs en fonction de i
}
```

dans les `()` on a 3 éléments :

1. `let i = 0` : on déclare le compteur,
2. `i < 5` : tant que i est < 5,
3. `i++` : augmente i de 1 à chaque tour.

Toujours le même principe, les critères entre `()` et le bloc exécuté entre `{}`

JavaScript vs Python

Aucun n'est meilleur que l'autre. Ils font des choses *différentes*.

- Vous voulez rendre une page web dynamique et y intégrer un jeu vidéo : Javascript
- Vous vouler jouer directement sur votre ordinateur : Python.

Ce qu'ils ont de commun :

- Ce sont des langages `de script`, ils sont `interprétés` et non compilés.

- Le typage est dynamique. Une variable peut désigner un entier puis une chaîne de caractère sans faire planter le programme.
- Ils sont orientés objets : `abc.def` (attribut `def` de l'objet `abc`)
- Ils sont considérés comme "faciles d'accès" par oppostion au C ou à Java, C++ etc.

Références

- Les cours sur le site et les parties suivantes
- w3schools
- MDN

De nombreux projets en javascript sont proposés aux élèves de terminale ISN dans la partie dédiée de mon site.