Introduction à Javascript

redgreenrefactor pour HETIC

PMD - 2020

Le Javascript

Le JavaScript est le premier langage de script pour le web. Il a été mis au point par Netscape en 1995. Ce langage de programmation est exécuté dans le navigateur d'un internaute et permet de dynamiser une page html. JavaScript nécessite un interpréteur pour être exécuté. Aujourd'hui, chaque navigateur possède son propre « interpréteur » ou « moteur » JavaScript

Lien MDN

CHARGEMENT D'UNE PAGE WEB

Connexion et demande du navigateur pour la page

Envoie du HTML

Demande des ressources additionnelles (JS, CSS...)

Retour des ressources

CLIENT

Exécution du JavaScript

SERVEUR

Exécution PHP, ASP et Java EE ...

H

Déclarative / impérative

HTML/CSS Programmation déclarative
JS Programmation impérative (séquences d'instructions exécutées)

Le JavaScript

Gestion des interactions entre l'utilisateur et le navigateur et évolution dynamique du contenu
Le JS peut manipuler le DOM
Document Object Model (représentation de la page)
Boucle et condition même principe que d'autres langages

Page dynamique

Interactivité

Réaction aux actions de l'utilisateur

Hétic formulaire

Autre exemple dynamique

- Accordion
- Datepicker
- Slider
- Carousel
- Autocomplete

Démo

Js Syntaxe

- Une instruction par ligne
- Fin de ligne par ; pas obligatoire MAIS … recommandé
- Donnée : variable ou on stock une information
- Traitement / Action / Instruction
- L'ordinateur fait ce qu'on lui demande
- Toujours déclarer une variable avant de l'utiliser

Affectation de variable

L'attribution d'une valeur à une variable se fait avec l'opérateur =

```
let temperature = 17.2;
```

Faire des calculs dans l'inspecteur

En pratique

Les types de variables

Les types de variables de base, sont sensiblement les mêmes dans tous les langages de programmation. Cependant des différences existent sur les détails et les traitements possibles sur ces types de variables.

Le type number

Désigne tous les nombres, entiers et décimaux.
Attention, on utilise la notation anglo-saxonne pour les nombres décimaux

```
let temperature = 17.2;
let age = 20;
```

Le type boolean

Un booléen est une donnée binaire, il ne peut avoir que deux états, logiquement opposés.
Une variable booléenne ne peut prendre que deux valeurs : true ou false.
Le sens exact de true et false dépendra du contexte.

```
let isCold = true;
let isHot= false;
```

Le type string

Une chaîne de caractère, string en anglais, est la version informatique d'un « bout de texte. » Ce peut-être une lettre, un mot, une phrase, un paragraphe ou n'importe quoi de représentable sous la forme d'une entité de type texte.

```
let question = "fait-il beau aujourd'hui?";
```

Le type array

Très utile pour stocker plusieurs valeurs dans une même variable. On utilise souvent les tableaux pour contenir une série d'éléments de même type, même s'il n'est pas obligatoire que tous les éléments soient du même type.

Exemple: une liste de fruits, d'élèves, de noms de planètes à afficher, de courses…

Les tableaux en javascript sont ordonnés, c'est-à-dire que leurs éléments sont indexés : à chaque case du tableau est assignée un index numérique, en partant de 0.

Le type array

Attention, chaque case de tableau peut contenir une variable de tout type (y compris un autre tableau).

```
// déclarer un tableau :
let bonbons = ["dragibus", "banane", "fraise tagada"];

// accéder à un élément du tableau avec la syntaxe [index] :
bonbons[0]; // => accède à "dragibus"
bonbons[2]; // => accède à "fraise tagada"
```

L'opérateur typeof

Le mot-clé typeof permet de vérifier le type d'une variable (ie. de son contenu) :

```
typeof 42;        //=> "number"
typeof 'Hello'; //=> "string"
typeof false;    //=> "boolean"
```

Faire des calculs dans l'inspecteur

En pratique

Balise script

On peut écrire du JS directement dans une page HTML entre des balises <script>

```
<script type="text/javascript">
    console.log('Hello world');
</script>
```

“Link” script

On peut écrire du JS dans une page à part comme pour le css

```
<script src="js/script.js"></script>
```

La concaténation

Concaténer 2 chaînes de caractères veut dire « les coller bout à bout. » L'ordre de la concaténation importe.
Pour concaténer deux chaînes, on utilise l'opérateur +

```
let nom = "world";
let message = "Hello " + nom;
// La variable message contient désormais la chaîne "Hello world"
```

Faire des calculs dans l'inspecteur

En pratique

Condition if

Si la condition est vérifiée, alors le bloc d'instructions associé est exécuté

```
if (condition) {
    // instructions si condition vérifiée
}
```

Condition if...else

Si la condition est vérifiée, alors le premier bloc d'instructions est exécuté, sinon le deuxième bloc est exécuté.

```
if (condition) {
    // instructions si condition vérifiée
}
  else {
    // instructions si condition non vérifiée
}
```

Condition if...else if...else

- Si la 1ère condition est vérifiée, alors le premier bloc d'instructions est exécuté ;
- Sinon, si la 2ème condition est vraie, alors le 2ème bloc est exécuté (note : cette 2ème condition n'est testée que si la première est fausse) ;
- ... On peut mettre autant de else if qu'on veut (note : au bout d'un moment, ça devient compliqué/illisible on fera alors plutôt un switch) ;
- Sinon si aucune des conditions précédentes n'est vraie, exécution du cas par défaut.

```
if (condition1) {
  // instructions si condition 2 vérifiée
}
else if (condition2) {
  // instructions si condition 1 non vérifiée, et condition 2 vérifiée
}
else {
  // instructions si aucune condition vérifiée
}
```

Les opérateurs de comparaison

= opérateur d'affectation (pour changer la valeur d'une variable, d'un élément de tableau…)

== opérateur de comparaison souple (ne compare que la valeur, pas le type)

=== opérateur de comparaison stricte (compare type & valeur)

Les opérateurs de comparaison

- `1 == '1';` => **true** : les types sont différents (number vs string), mais la valeur est considérée être la même
- `1 === '1';` => **false** : la valeur est la même, mais les types sont différents
- `1 == 1;` => **true** : la valeur est la même (et le type aussi, mais ce n'est pas vérifié)
- `1 === 1;` => **true** : la valeur et le type sont les mêmes, tout va bien

Les opérateurs logiques

- L'opérateur && signifie "ET"
- L'opérateur || signifie "OU"

Objectifs pour la prochaine séance

Jeu du plus petit ou plus grand

Instruction

1. Au chargement de la page, on va générer un nombre entier aléatoire entre 1 et 500 (C'est le nombre que va devoir deviner le joueur)
2. On demande au joueur de rentrer un entier dans une boîte de dialogue.
3. On indique au joueur via un popup si le nombre à trouver est plus petit ou plus grand.

Bonus

- Tant qu'on n'a pas trouvé, on revient au 2.
- Lorsque le joueur trouve le bon entier, ça affiche un popup célébrant sa victoire.
- Si le joueur n'a pas trouvé au bout de 10 tentatives, on affiche un message de défaite.

Bonus ultra

- Ajout d’une véritable interface HTML et CSS
- Enregistrement de scores

Un exemple vous sera donné dans l’extranet

Les liens utiles

- Number
- Math.random
- Math.Floor
- If...else

- While
- Do...while


MOTIVATION