



Support de cours JavaScript

FRÉDÉRIC NADARADJANE

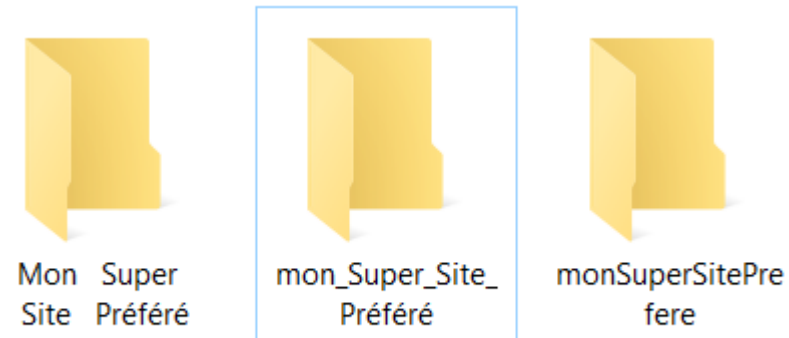
LA GRANDE CLASSE

30/03/2023

0.Rappels

Rappels sur HTML / algorithmie

- Quel balise permet de faire un titre en HTML ?
- Quel est la meilleure façon de faire du css :
- Créer un nouveau fichier css
- Mettre le style de la page dans une balise <style>, à l'intérieur du fichier html ?
- Je dois créer un nom de dossier : lequel parmi ces propositions est un nom correct :
- Mon Super Site Préféré/
- monSuperSitePrefere/
- mon_Super_Site_Préfééré/



Rappels sur HTML / algorithmie

- Différence entre client et serveur ?
- Pour chaque langage dire si c'est client ou serveur.
- PHP ?
- HTML ?
- MYSQL ?
- JavaScript ?
- Comment faire une condition en algorithmie ? En PHP ?

Rappels sur HTML / algorithmie

Exercice : 10 min

- Crée une page HTML avec un titre. Celui sera en centré, en gras, rouge et souligné.
- En dessous il aura un bloc de texte (lorem ipsum) dans une balise p.
- **Attention la feuille de style doit obligatoirement être dans une page différente !!!!**

1.Introduction

Présentation

- Javascript a été lancé par Netscape en 1995
- Premier langage à avoir été développé pour le web et demeure aujourd'hui comme le plus répandu.
- Permet d'étendre les possibilités du HTML
- Très souvent utilisé pour ajouter de petites fonctionnalités aux site Web.

Présentation

- Langage de script objet
- Syntaxe style C/C++/JAVA
- N'est **PAS** du JAVA
- Exécuté par le client Web
- Peut être désactivé par le client
- Nombreux objets pour la manipulation HTML
- Gestion des événements HTML
- Rendre les pages dynamiques (HTML + CSS + JS)

Présentation

- Langage interprété (\neq compilé)
- Code peu typé.
- Code lisible directement depuis le navigateur (\neq confidentialité)
- Sensible à la casse (ex: fonction Bonjour() \neq fonction bonjour()).

2. Les bases

Les bases Intégration

Dans la page HTML :

- Dans la balise <head>
- A la fin de la balise <body> (optimisation du chargement de la page)

Exemple :

```
<script>
```

```
    //Mon code JavaScript
```

```
</script>
```

Les bases Intégration

En fichier externe (recommandé) :

- **Similaire à l'intégration dans la page HTML**

Exemple :

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

Les bases

Syntaxe générales

Le JavaScript fonctionne par instructions, chaque instruction est exécuté de façon linéaire :

- Chaque instructions se terminent par un point-virgule
- Les différents espaces ne sont pas traités (entre les opérateurs par exemple)

L'indentation est autorisée et vous permet d'avoir un code lisible :

- Tabulation
- Indentation de 2 espaces
- Indentation de 4 espaces

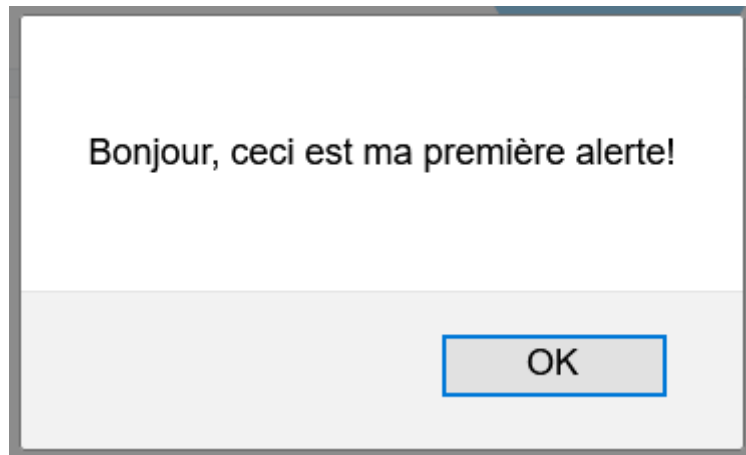
Les bases

Méthode alert()

Une fonction de base est « alert() », elle permet de créer une fenetre de dialogue à l'utilisateur.

Exemple :

```
alert("Bonjour, ceci est ma première alerte!");
```



Les bases

Les commentaires

JavaScript offre la possibilité de commenter sur une ligne ou en multiligne :

Commentaire sur une ligne : "//"

Exemple :

```
// Ceci est un commentaire
```

Commentaire multiligne : "/*" pour ouvrir, "*/" pour fermer

Exemple :

```
/*
```

```
    Ceci est un commentaire multiligne
```

```
*/
```

Les bases

Déclaration d'une variable

Syntaxe :

`var` [#NOM_DE_LA_VARIABLE#]

Contraintes :

- Les noms doivent commencer par une lettre
- Les noms peuvent également commencer par \$ et _
- Les noms sont sensible à la casse (y et Y sont des variables différentes)
- Les mots réservés comme les mots clés JavaScript) ne peuvent pas être utilisés comme noms

Les bases

Déclaration d'une variable

Déclaration d'une variable simple :

```
var myVar;
```

Déclaration de plusieurs variables :

```
var myVar, myOtherVar, myLastVar; (CamelCase)
```

Déclaration avec attribution de valeur :

```
var intVar = 3;
```

```
exStringVar = "Message";
```

Les bases

Déclaration d'une variable - Type

Le type des variables n'a pas besoin d'être spécifier explicitement, on parle de type dynamique en JavaScript. La langage va lui-même définir un type selon la valeur qu'on lui définit.

Quelques types simples :

- Les nombres
- Les chaînes de caractères
- Les booléens

Les bases

Déclaration d'une variable - Nombre

Déclaration d'un nombre entier :

```
var nbVar =1;
```

Déclaration d'un nombre à virgule flottante :

```
var floatVar = 42.54; (Attention : c'est bien un point et non une virgule)
```

Déclaration de plusieurs nombres entier :

```
var nbVar = 1, nbVar = 2.342, nbVar = 34;
```

Les bases

Limites des nombres

Attention aux flottants : les nombres flottant sont exactes jusqu'à 15 chiffres :

```
var x = 9999999999999999 //Va donner 9999999999999999
```

```
var y = 9999999999999999 //Va donner 10000000000000
```

Les bases

Déclaration d'une variable - String

Déclaration d'une chaîne de caractères :

```
var strVar = "Message"; //double quotes
```

Ou

```
var strVar = 'Message'; //simple quotes
```

Remarque : Les deux notations sont équivalentes.

Déclaration de plusieurs chaînes de caractères:

```
Var strVar ='a', strVar2 ='à', strVar3 ="Test d'une chaîne"
```

Les bases

Exercices:

Créer un fichier hello.html et un fichier hello.js. Appeler le fichier hello.js dans le fichier html.

1. Dans le fichier hello.js créer une variable avec le contenu "Ma première variable".
2. Créer une alert avec le contenu "Hello world !"

Les bases

Déclaration d'une variable - String

Pour une meilleure lisibilité, éviter les lignes de code de plus de 80 caractères.

- Si une instruction JS ne correspond pas à une ligne, le meilleur endroit pour le casser est après un opérateur.

```
var strVar =
```

```
"Lorem ipsum, dolor sit amet consectetur adipisicing elit";
```

- Si votre texte est vraiment grand, préférer concaténer :

```
var =
```

```
"Lorem ipsum, dolor sit amet consectetur adipisicing elit."+
```

```
"Quisquam culpa inventore consectetur reprehenderit quae,"+
```

```
" vitae accusantium fugiat vitae accusantium fugiat";
```

Les bases

Déclaration d'une variable - String

- Dans le cas où vous voulez intégrer des quotes simples dans vos chaînes de caractères, les déclarer avec les quotes doubles (et inversement) :

```
var strVar3 = "Lorem ipsum, dolor 'sit amet' consectetur adipisicing elit.";
```

```
var strVar4 = 'Lorem ipsum, dolor "sit amet" consectetur adipisicing elit.'
```

- Si cela ne suffit pas, il est possible d'échapper les caractères avec l'anti-slash "\" :

```
var strVar3 = "Lorem ipsum, dolor \'sit amet\' consectetur adipisicing elit.";
```

```
var strVar4 = 'Lorem ipsum, dolor \'sit amet\' consectetur adipisicing elit.'
```


Les bases

Concaténation

- Il est possible de concaténer différentes chaînes de caractères:

```
var text = "Hello" + " " + "World!";
```

ou

```
var text = "Hello".concat(" ", "World!");
```

- On peut aussi concaténer des nombre à des chaînes de caractères :

```
var strVar = "Hello", nbVar = 6, concatVar //Déclaration des variables
```

```
concatVar = strVar +nbVar;
```

Résultat : "Hello6"

Les bases Booléen

Les booléens sont utiles pour différents tests :

Déclaration d'un booléen :

```
var boolOk = true;
```

ou

```
var boolKo = false;
```

Déclaration de plusieurs booléens :

```
var bool1 = true, bool2=false, bool3;
```

Les bases

Déclaration d'une variable - Tableau

Les tableaux JavaScript sont utilisés pour stocker plusieurs valeurs dans une seule variable.

Syntaxe : *var arrayName = [item1, item2, ...];*

Exemple :

```
var cars = ["Audi","Mercedes","BMW"]
```

Les espaces et les sauts de ligne ne sont pas importants. Une déclaration peut s'étendre sur plusieurs lignes :

```
var cars = [  
    "Audi",  
    "BMW"  
]
```

Correction

Les bases

Récupérer le type d'une variable

Pour récupérer le type d'une variable, JavaScript nous fournit la fonction "typeof()".

Astuces:

- Au lieu d'afficher plusieurs boîtes de dialogues, il est possible de concaténer les différents éléments
- Le caractère "\n" permet de faire un retour à la ligne

Les bases

Récupérer le type d'une variable

Exemple :

// Déclaration

var nbVar =1, strVar = "Hello", boolOk = true;

// affichage

```
alert ("Nombre: " + nbVar+" de type: " + typeof(nbVar)+"\nChaîne de caractère:"+strVar+  
|      "de type:" + typeof(strVar) + "\nBooleen :"+boolOk+"de type:" + typeof(boolOk)  
);
```

Les bases

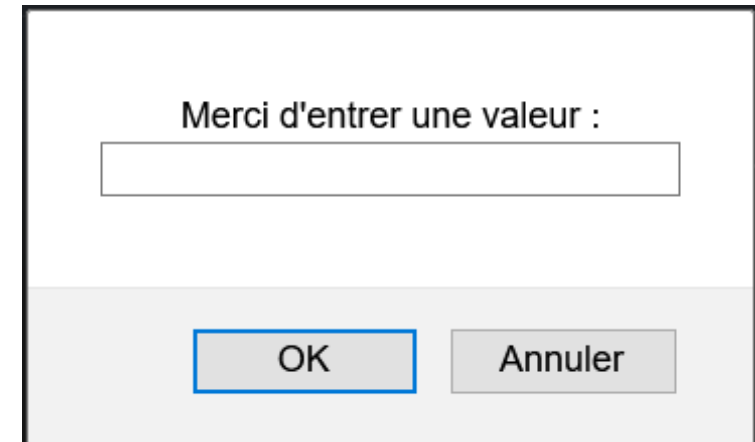
Faire interagir l'utilisateur

Pour clore cette partie, voici 2 méthodes permettant à l'utilisateur d'interagir :

- **prompt()** : permet à l'utilisateur de saisir une chaîne de caractère qui sera stocké dans une variable
 - Reçoit en premier paramètre une chaîne de caractère à afficher pour informer l'utilisateur, similaire à "alert()".
 - Utiliser "prompt()" comme étant la valeur d'une variable

Exemple:

```
var valueUser =prompt("Merci d'entrer une valeur :");
```



Merci d'entrer une valeur :

OK Annuler

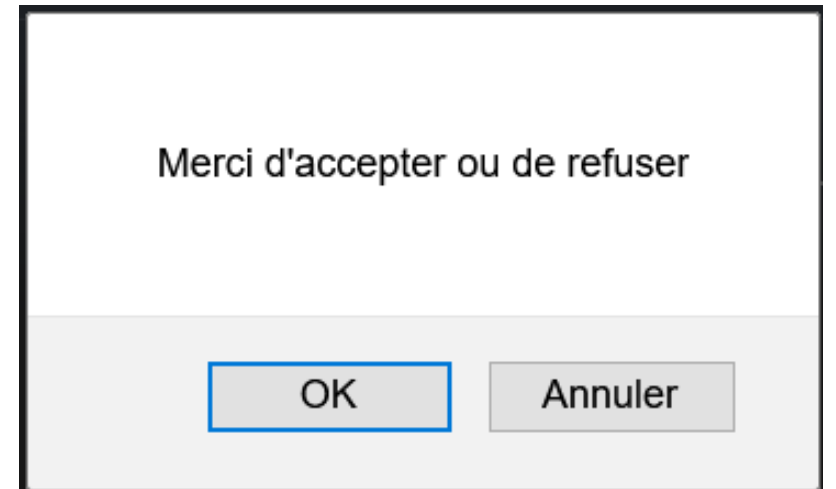
Les bases

Faire interagir l'utilisateur

- **confirm()** : affiche une boîte de dialogue, propose deux boutons "Ok" et "Annuler". Cette méthode retourne un booléen en conséquence.
 - Reçoit en premier paramètre une chaîne de caractère à afficher pour informer l'utilisateur, similaire à "alert()".
 - Utiliser "confirm()" comme étant la valeur d'une variable

Exemple:

```
var boolUser =confirm("Merci d'accepter ou de refuser");
```



Les bases

Exercice :

Demandez le nom de l'utilisateur et affichez "bonjour" suivit de son nom dans une alert()

Les bases Debugger

Quelques pistes pour déboguer un script JavaScript :

- Examineur d'élément => onglet "console" :
 - ❖ Permet d'accéder au debugger d navigateur, vous pouvez donc accéder à vos erreurs en direct
 - ❖ Vous pouvez taper directement du JavaScript dans la console si besoin
- Examineur d'élément => onglet "Débogueur"
 - ❖ Permet d'accéder à votre script
 - ❖ Permet de placer des points d'arrêts
- `console.log()`: vous permet d'afficher vos variables dans la console du navigateur

Les bases

Comment afficher mes retours JS sans le HTML ?

On anticipe un peu sur le cours pour pouvoir nous permettre de tester.

- `document.write(#data#)` : permet d'afficher du contenu dans le HTML

Exemple :

```
document.write('bonjour!');
```

- Mettons tout cela en application :
 1. Créer une div dans le HTML portant l'id demo (`<div id="demo"></div>`)
 2. Vous pouvez alors appeler cette instructions dans votre script JavaScript pour afficher vos données dans cette div :

```
document.getElementById("demo")
```

Les bases

Comment afficher mes retours JS sans le HTML ?

○ Mettons tout cela en application :

1. Créer une div dans le HTML portant l'id demo (<div id="demo"></div>)
2. Vous pouvez alors appeler cette instructions dans votre script JavaScript pour afficher vos données dans cette div :

```
document.getElementById("demo").innerHTML = "Hello World !";
```

Remarque importante : dans mes différents exemples j'utilise la convention de nommage "#quelque chose#". Il faut bien sûr **remplacer la valeur** en question à l'intérieur des dièses ET **retirer les dièses**.

3.Manipuler les types simples

Manipuler les types simples

Opérations arithmétiques

- Opération arithmétiques :

Balise	Description
+	Addition
-	Soustraction
*	Multiplication
/	Division
%	Modulo : reste de la division entière

Manipuler les types simples

Opérations arithmétiques

- Raccourcis d'opération arithmétiques :

Balise	Description
<code>+=</code>	Addition
<code>-=</code>	Soustraction
<code>*=</code>	Multiplication
<code>/=</code>	Division
<code>%=</code>	Modulo : reste de la division entière

Exemple avec l'addition :

```
var nb1 = 5;
```

```
nb1 = nb1 + 6;    // forme classique
```

```
nb1 += 6;         // forme raccourcie
```

Manipuler les types simples

Opérations arithmétiques

- Incrémentation et décrémentation:

Balise	Description
++var	Pré incrémentation
var++	Post incrémentation
--var	Pré décrémentation
var--	Post décrémentation

Exemple avec l'addition :

```
var nb1 = 5;  
nb1 = nb1 + 1;    // forme classique  
nb1++;            // forme raccourcie
```


Manipuler les types simples

Opérations arithmétiques

Différence entre pré et post :

Pré-incrémentation :

- Etape 1 : opération sur la valeur
- Etape 2 : affichage de la valeur

Post-incrémentation :

- Etape 1 : affichage de la valeur
- Etape 2 : opération sur la valeur

Exemple (à tester pour bien comprendre):

```
var number = 0;  
alert(++number);  
alert(number);  
alert(number++);  
alert(number);
```

Manipuler les types simples

Chaîne de caractères

- **Length** : Propriété permettant d'obtenir le nombre de caractères de la chaîne de caractère
- **toLowerCase()**: mise en minuscule de la chaîne de caractères
- **toUpperCase()**: mise en majuscule de la chaîne de caractères
- **charAt(#position#)**: retourne le caractère situé à une position précisée (la première position est "0")

Manipuler les types simples

Chaîne de caractères

Exemple :

```
var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
var sln = txt.length;
```

```
var text1 = "Hello World!";           // String  
var text2 = text1.toLowerCase();       // hello world!  
var text3 = text1.toUpperCase();       // HELLO WORLD!  
text1.charAt(0);                       // returns H
```

Manipuler les types simples

Chaîne de caractères

Exercice :

Crée une balise h1 avec le texte "bonjour" et mettez le en majuscule grâce à JavaScript

Crée un prompt, et envoyez une alert avec le nombre de caractères que compose le mot entré par l'utilisateur.

4.Structures de base

Structure de base

Opérateur de comparaison

- Opérateur de comparaison :

`==` : égal

`!=` : différent de

`>` : supérieur à

`<` : inférieur à

`>=` : supérieur ou égal à

`<=` : inférieur ou égal à

`===` : valeur égales avec même type

`!==` : valeur différentes avec même type

Ces opérateurs de comparaison renvoient un booléen

Structure de base

Opérateurs logiques

- Opérateur logiques:

&& : et

|| : ou

! : opérateur de négation (not)

Structure de base

Structure conditionnelle – if, else, else if

Lorsque nous voulons tester la valeur d'une variable, on peut utiliser une structure conditionnelle.

Structure :

if (#condition #) : si la condition est vrai

else : sinon

Else if (# condition #) : sinon si une autre condition est vrai

Exemple :

```
if (time < 10) {  
    greeting = "Good morning";  
} else if (time < 20) {  
    greeting = "Good day";  
} else {  
    greeting = "Good evening";  
}
```


Structure de base

Structure conditionnelle – if, else, else if

Astuce pour réduire l'écriture :

- Si vous effectuez un "if" sur un booléen, vous pouvez omettre l'égalité.

Cela viens du fait que la condition testée par le "if" est toujours testée par rapport à "true"

Exemple :

```
var boolOk = true
```

```
if(boolOk){
```

```
.....
```

```
}
```

Structure de base

Structure conditionnelle – if, else, else if

Exercice : Site recensement

Demander à l'utilisateur son année de naissance. Si la personne à plus de 16 ans, on affichera une alerte lui demandant de faire son recensement. Sinon, on lui affichera dans combien de temps il devra la faire.

Structure de base

Structure conditionnelle – if, else, else if

Exercice : Politique acceptation de cookies

Demander à l'utilisateur d'accepter la politique de sauvegarde de données. Si l'utilisateur accepte, on affichera une alert en lui souhaitant la bienvenue sur le site. Sinon on affichera une alert en lui demandant de quitter de façon IMMEDIATE le site.

Structure de base

Structure conditionnelle – if, else, else if

Exercice

Demander à l'utilisateur son âge.

Créer une autre alert, dans lequel on affiche l'âge rentré par l'utilisateur et on demande de le confirmer.

Si l'utilisateur le confirme, on affiche une alert qui remercie l'utilisateur d'avoir rentré son âge.

Si l'utilisateur ne confirme pas, on l'invite à ressaisir son âge.

Afficher l'âge sur le document html

Structure de base

Structure Itérative -while

On utilise une boucle pour exécuter un bloc d'instruction plusieurs fois selon une condition. Il existe 4 types de boucles.

While :

La boucle while exécute un bloc tant qu'une condition spécifiée est respectée.

Exemple :

```
while (i < 10) {  
    text += "The number is " + i;  
    i++;  
}
```

Structure de base

Structure Itérative -while

Do while :

La boucle do while est une variante de la boucle while. Cette boucle exécute le bloc de code une fois, avant de vérifier si la condition est vraie, puis elle répétera la boucle tant que la condition est vraie.

Exemple :

```
do {  
    text += "The number is " + i;  
    i++;  
}  
while (i < 10);
```

Structure de base

Structure Itérative - while

Demandez à l'utilisateur son âge tant son âge est inférieur à 12, affichez le prompt

Structure de base

Structure Itérative - For

For :

La boucle for permet d'exécuter un bloc d'instruction un certain nombre de fois selon une ou plusieurs variables qu'on incrémente jusqu'à ce qu'elles ne respectent plus une conditions donnée.

For (initialisation; condition ; incrémentation)

Exemple :

```
for (i = 0; i < 5; i++) {  
    text += "The number is " + i + "<br>";  
}
```


Structure de base

Structure Itérative - for

Afficher 10 fois le nom du client sur la console entré sur le prompt par l'utilisateur.

Structure de base

Rappels

Dire quelle boucle sera la plus adaptée :

Faire un contrôle sur une saisie faite par l'utilisateur

Afficher les 10 meilleurs articles sur une page

Afficher un message rouge jusqu'à ce que l'utilisateur entre au minimum 4 lettres

5.Fonctions utilisateur

Fonctions utilisateur

Présentation

Les fonctions utilisateurs permettent au développeur de créer ses propres fonctions.

Syntaxe :

```
Function myFunction(arguments){  
  // Le code que la fonction va devoir exécuter  
}
```

Fonctions utilisateur

Portée des variables

Les variables déclarées hors des fonctions sont dites "globales" et accessibles à tout endroit du code.

Les variables locales sont les variables créées à l'intérieur d'une fonction et sont détruites à la sortie de la fonction.

Remarques:

- La portée d'une variable locale est donc limitée à sa propre fonction et n'est pas accessible depuis l'extérieur
- En cas de déclaration d'une variable locale dans une fonction du même nom qu'une variable globale, la variable globale ne sera pas affectée

Fonctions utilisateur

Les arguments – simple, multiples

Il est possible qu'une fonction possède des arguments qui serviront au traitement de la fonction.

Exemple argument simple :

```
function calculTaux(prixInitial){ .....}
```

Exemple argument multiples :

```
function calculTaux(prixInitial, produit){.....}
```

Fonctions utilisateur

Les valeurs par défaut des arguments

En JavaScript, par défaut, la valeur des paramètres d'une fonction sera undefined. Malgré tout, il peut être assez utile de pouvoir définir d'autres valeurs par défaut.

Grâce aux paramètres par défaut qui existent depuis ECMAScript 2015(ES6), on peut se passer de cette vérification et alléger le code de la fonction :

Exemple :

```
function calculTaux(prixInitial = 5.50){  
  
    ....  
  
}
```

Exercices

Crée une fonction, qui demandera son age. Si il est supérieur à 18 le site lui affiche une alert :
« Vous êtes majeurs, vous débloquez toutes les fonctionnalités du site » à l'inverse, le site renverra une alert :

« Vous êtes mineur, le site sera partiellement bloqué »

L'âge sera entrer par l'utilisateur et sera passé en paramètre.

Exercices

Crée une fonction, qu'on appellera `calculSolde`, permettant de calculer le prix final de l'article payé en euros ainsi que la remise effectuée en euros. Le prix initial, la remise sera entrer par l'utilisateur et sera passé en paramètre