JAVASCRIPT

L.Freund WebForce3

Des ressources

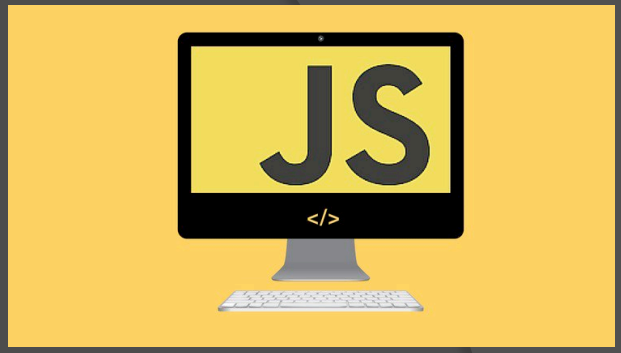
- http://silentteacher.toxicode.fr/
- codecombat
- https://www.codecademy.com/learn/learn-javascript
- https://www.codeschool.com/learn/javascript

- http://slides.com/lior-chamla/javascript#/
- http://slides.com/lior-chamla/deck#/21
- Freecodecamp
- https://sutterlity.gitbooks.io/apprendre-jquery/index.html

A quoi ça sert ?

- Rajouter du dynamisme dans une page

Introduction



- Javascript != Java
  - Javascript s'exécute sur le Client (navigateur)
  - Servlet en Java: s'exécute sur le serveur
- Exécution sans rechargement de page
  - -> rapidité, meilleur UX
- Concurrents : Flash Player, applet Java

Les outils

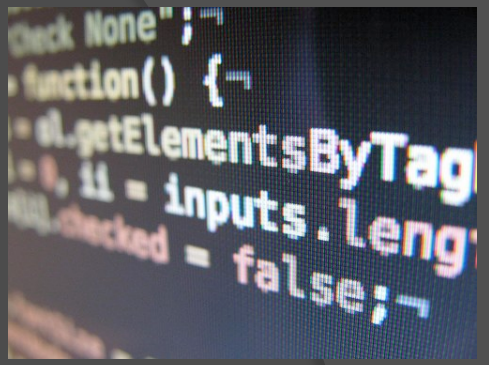
- Editeur : sublimeText, brackets, …
- Navigateur avec débuggeur
  - Chrome, Firefox…
- Documentation de référence: w3school javascript reference

La console Javascript

- Pour ouvrir la console
  - Vue/développeur/Console JavaScript
    - F12 / (Alt-Cmd-i)

Ou placer le code Javascript ?

Ou placer le javascript ?



1. Directement dans le html

```
<script> codeJS </script>
```

2. Dans l'action d'un button : Event

```
<button onclick="codeJS">Ok</button>
```

3. Dans un fichier.js à part

```
<script src="js/tp1.js"></script>
```

1) Directement dans .html

- <script> *codeJS* </script>
  A la fin du body (plutôt que dans le head) pour accélérer exécution

```
<!DOCTYPE html>
<html>
<head>
        <title> Javascript insde HTML </title>
        <meta charset="utf-8">
</head>
<body>

<h1>Petit Test directly in HTML</h1>
<p id="txt"></p>
…
<script>
        document.getElementById("txt").innerText = "WF3";
</script>
</body>
</html>
```

2) Fichier à part : tp1.html+ js/tp1.js

```
<!-- tp1.html -->
<!DOCTYPE html>
<html>
<head>
        <title> TP1 Javascript </title>
        <meta charset="utf-8">
</head>
<body>

<h1>My First JavaScript</h1
…


…
<script src="js/tp1.js"></script>
</body>
</html>
```

2) Fichier à part : tp1.html+ js/tp1.js

```
/*
 js/tp1.js
 petit essai JS
*/
console.log( "Start JS");
```

3) Dans l'action d'un button (Event)

// technique deprecated

```
<!DOCTYPE html>
<html>
<head>
        <title> Javascript inside HTML </title>
        <meta charset="utf-8 »>
</head>
<body>

<h1>My First JavaScript in HTML</h1>
<p id="idTxt"> Abracadabra… </p>
<button
 onclick="document.getElementById('idTxt').innerText='Le Lapin!'">Tour de Magie
</button>
</body>
</html>
```

Agir sur le HTML en JS

Principe

```
<p id="idTxt"> Abracadabra… </p>
```

1) Récupération d'un élément à partir de son id
   `getElementById()` …
2) Modification d'une propriété de l'élément
   innerText,.innerHtml, .src, .style…

Tous les éléments ont
- un ensemble de propriétés communes (+fonctions)
https://www.w3schools.com/jsref/dom_obj_all.asp
- un ensemble de propriétés propres (exemple img)
https://www.w3schools.com/jsref/dom_obj_image.asp

Exemple changement texte

```
<p id="idTxt"> Abracadabra… </p>


1) Récupération d'un élément à partir de son id
                                        …
document.getElementById('idTxt')
…
2) Modification d'une propriété de l'élément

document.getElementById('idTxt').innerText='Le Lapin !!!';


(bis) Le même avec mise en forme
document.getElementById('idTxt').innerHTML='Le <b>Lapin</b> ';
```

Changement propriété image

```
<img id="idImg">

…

                                    …
document.getElementById('idImg').src='bob.png';
```

Changement d'autre propriété d'une image : https://www.w3schools.com/jsref/dom_obj_image.asp

Changement style css

- Changement de propriétés

```
document.getElementById('idTxt').style.fontSize='25px';
document.getElementById('idTxt').style.color='red';
document.getElementById('idTxt').style.backgroundColor='red';
// ATTENTION backgroundColor au lieu de background-color
```

- Changement de la propriété: display

```
// cacher
document.getElementById('idTxt').style.display='none';
// afficher
document.getElementById('idTxt').style.display='block';
```

Exemple sympa : LED

```
<!DOCTYPE html>
<html>
<head>
        <title> TP2 Led </title>
        <meta charset="utf-8">
</head>
<body>

<h1>Jour-Nuit…</h1>

<img id="idImg" src="led-off.png">
<p>
<button onclick="document.getElementById('idImg').src='img/led-on.png'">On</button>
<button onclick="document.getElementById('idImg').src='img/led-off.png'">Off</button>

</body>
</html>
```

TP

Mettre un bouton [On/off] qui allume/éteint la Led

Ajouter 1 bouton [+] et 1 bouton [-]

Faire que quand on appuie sur [-] la led disparaît et [+] la fait apparaître.

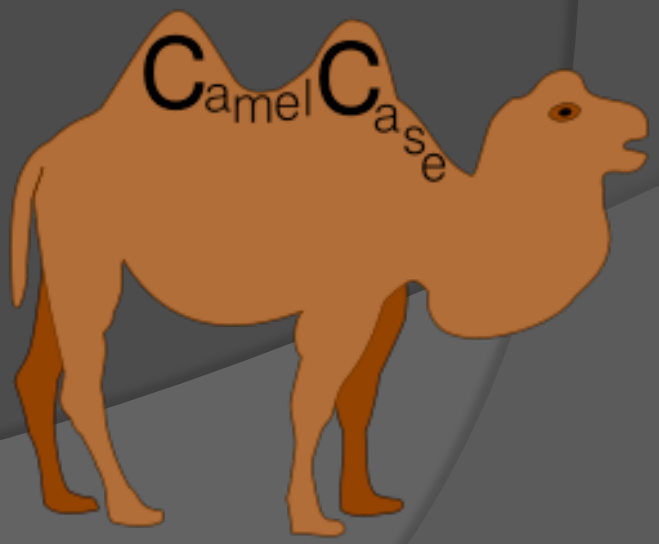
Questions subsidiaires

- Rajouter un bouton pour changer de couleur de led
- Changer le texte du bouton On/off

Fondamentaux de la programmation

Syntaxe

- Commentaires
  - // 1 ligne
  - /* plusieurs lignes */
- Indentation
  - OBLIGATOIRE
- Nommage: Camel Case
  - maVariable
  - maFonction
  - MaClasse



Les variables locales

- Déclaration non typé

```
var prenom;
        // Attention : non initialisé -> undefined dans la console

var nom = "Rosita"; // initialisation
var indicePollution, i=0 ;
```

- Initialisation/affectation

```
nom = "Morane";
prenom = "Bob";
indicePollution = 12;
```

- CaseSensitive, camelCase

Les différents types

```
var i = 12 ; // number
var nom = "Rosita"; // string
var visible = true ; // boolean
```

- Tableaux

```
var prenoms = ["Rosita", "Bob", "Gaston"]; // Tableau
var ages = [12, 24, 36]; // Tableau
```

- Objet

```
var personne= {prenom:"Laurent", nom:"Freund"}; // Objet
```

- Verification du type d'une variable en utilisant typeof

```
typeof nom // string
typeof i // number
typeof visible // boolean
typeof prenoms // object
typeof personne // object
```

Les variables undefined

```
var tmp ; // undefined
```

- La valeur de tmp est undefined
- Le type de tmp est undefined
- `tmp = 2 ; // tmp devient un number`

- On peut vider/dé-initialiser une variable en la déclarant undefined

  ```
  tmp = undefined ;
  ```

- Pour un objet on utilisera null qui ne met pas le type à undefined (reste object)

  ```
  var personne = null ;
  ```

Les tableaux

```
var tabPrenoms = ["Rosita","Bob","Gaston"]; // Tableau
var tabNoms = []; // Tableau

Le 1er élément ("Rosita")est dans la case 0, le dernier
dans la case : nb de cases-1

prenom = tabPrenoms[0]; // case 1 ;
prenom = tabPrenoms[1]; // case 2 ;
prenom = tabPrenoms[2]; // case 3 ;

tabNoms.push("Roberto"); // ajout fin
tabNoms[tabNoms.length] =  "Roberto"; // idem

tabNoms.pop() ; // suppression dernier elt
delete tabNoms[1] ; // suppression elt 1 -> undefined
```

Les chaines de caractères

- Simple cote ou double cote

```
var prenom = "Bob";
var nom = ‘Gaston’;
```

- Concaténation

```
var nomComplet = prenom + " " + nom;
var i=3 ;
nomComplet = prenom+i ;
```

Affichage de contenu/variable

Affichage

3 affichages différents en fonction des besoins

- Dans la console
- Dans une popup window
- Dans le html
- Dans des éléments HTML

Exemple : tp1.html + affichage.js

```
<!DOCTYPE html>
<html>
<head>
        <title> TP2 Javascript Affichage </title>
</head>
<body>

<h1>Exemple d'affichage</h1>
Console + window +modif html

<p id= "txtHello"> <p>
<p id= "txtNom"> <p>

<script src= "js/affichage.js"></script>
</body>
</html>
```

affichage.js

```
/* Affichage des contenus des variables console */
var prenom = "bob";

console.log("Hello WF3");
console.log(prenom);
```

affichage.js

```
/* Affichage des contenus des variables popup */

var prenom = "bob";

window.alert("Hello WF3") ;
window.alert(prenom) ;
```

affichage.js

```
/* POUR DEBUG Affichage des contenus directement dans html  */

var prenom = "bob";

document.write("Hello WF3");

// ATTENTION, efface tout si page
entièrement loadée
```

```
<button onclick="document.write(5 + 6)">ATTENTION !!!</button>
```

affichage.js

```
/* Affichage des contenus directement dans les éléments html */

var prenom = "bob";


document.getElementById("txtHello").innerText = "Hello WF3";
document.getElementById("txtNom").innerText = prenom;
```

Saisie d’une valeur

Ouverture d’une Pop-up de saisie

```
var val=prompt("Combien ?") ;
```

Variable globale

On ne met pas le mot var

```
patron="Robert";
patronne="Rita";
```

Opérateurs

- +, -, * /, % (modulo)
- ++, --, +=, -=, *=, /=

- ex :

```
i=i+1;
i= i*2 ;
i++ ;
i = 2+10*4 ;
i = (2+10)*4 ;
i +=4 ;
i *=3 ;
```

Comparaison

```
if(a==b)
…
if(a===b) // verifie également que les types sont identique : A UTILISER
…

if(a > b)
…
if(a >= b)
…

if(a != b) // n'est pas égal
if(a !== b) // n'est pas égal ou type différents

…

if(a===b || a===c) // or
…

if(a===b && b= == c) // and
…
if(!(a === b)) // ! : contraire
```

if/else

```
if(…) {// 1 seul cas
}

if(…) {// 2 cas possibles
…
} else {
…
}

if(…) {
…
} else if(…) {) // si besoin d'un autre cas…
…
} else if(…) {) // si besoin d'un autre cas…
…
} else { // si aucun des cas
…
}
```

Exemple if

```
var date = new Date() ;
var heure= date.getHours() ;
if(heure===12) {
        window.alert("Midi");
} else if(heure===24) {
        window.alert("Minuit");
} else
        window.alert( heure+" h");
```

Bibliothèque extérieure

Utilisation de l'objet Date

Récupérer la date

```
var d = new Date();
var y = d.getFullYear() ;
var m = d.getMonth() ;
var dStr = d.toLocaleTimeString() ;
```

TP

- Faire une page qui affiche la date d'aujourd'hui
- Qui affiche bonne année si on est en janvier
- Qui affiche bon anniversaire laurent si on est le 3 janvier et tous les anniversaires des WF3.
- Qui affiche une image de fond de nuit si on est entre 20h et 7h
- Qui affiche une image de jour … le jour

Les boucles

Les boucles for/while/ do while

- for(…){

```
for(var i=0 ; i<3 ; i++) {
    window.alert("I="+i);
  }
```

Boucles et tableaux

1ère possibilité

```
for(var i=0 ; i < tabNoms.length ; i++)
      … // do something interesting
```

2nd possibilité

```
for(i in tabNoms)
      … // do something interesting
```

Les boucles for/while/ do while

- while(…){

```
var i=0 ;
while(i<3) {
     window.alert("I="+i);
     i++ ;
  }
```

!!! Attention aux boucles infinies

Les boucles for/while/ do while

- do… while(…);

```
var i=0 ;
 do {
    window.alert("I="+i);
    i++ ;
 } while(i<3) ;
```

!!! Attention aux boucles infinies

La gestion des erreurs

- try… catch(…);

```
try{
… //je fais plein d'erreurs
   youplaboum ;
  window.alert("Hello");
} catch(erreur){
      // je capture l'erreur et la gère
      window.alert(erreur.message);
      throw("Stop !!!");
  }
  window.alert( "Et ça continue encore et …");
```

TP

- Faire un tableau contenant les noms de 4 personnes
- Faire un tableau contenant les ages de 4 personnes
- Afficher dans le HTML les noms et ages des 4 personnes
- Afficher le nom et l'age du plus agé
- Afficher le nom et l'age du moins agé
- Afficher l'age moyen

Les fonctions

Les fonctions

- Permet de ne pas réecrire du code
- Lisibilité
- Copier/coller = erreur de conception
- prend des entrées (optionnel) : paramètres d'entrée de la fonction
- génère une sortie (optionnel) : paramètre de sortie de la fonction (return)

2 déclarations possibles

```
function nomFonction(){
        … // je fais des trucs passionnants
}
```

Ou fonction anonyme

```
var nomFonction = function() {
        … // je fais des trucs passionnants
}


// je l'appelle pour faire des trucs…
nomFonction();
```

Définition + utilisation fonction

Fonctions avec paramètres d'entrées

```
/* defº fctº passionnante */
function nomFonction(param1, param2…){
    … // je fais des trucs passionnants
}

// je l'appelle pour faire des trucs…
nomFonction(param1, param2…);
```

Exemple

```
/* petit test de fonction */
function afficheAge(nom, annee){
    var age ;
    age = 2017-annee ;
    console.log(nom+" a "+age+" an(s)");
}

afficheAge("Bob",2000);
afficheAge("Rosita",1980);
```

Définition + utilisation fonction

Fonctions avec paramètres d'entrées
+ paramètre de retour

```
/* defº fctº passionnante */
function nomFonction(param1, param2…){
        var val ;
        … // je fais des trucs passionnants
        return val ;
}

// je l'appelle pour faire des trucs…
var v1 ;
v1=nomFonction(param1, param2…);
```

Exemple

```
/* petit test de fonction */
function calcMoy(nb1, nb2){
     var moy ;
     moy = (nb1+nb2)/2 ;
     return moy ;
}

var m1, m2 ... ;
m1 = calcMoy(1000,2000);
m2 = calcMoy(3455, 2);
...
```

gros-exemple-fct.html

```
<!DOCTYPE html>
<html>
<head>
          <title> Les fonctions </title>
          <meta charset="utf-8">
          <meta name="description" content="">
          <meta name="keywords" content="">
</head>
<body>
<h1>TP sur les fcts avec return</h1>
Ceci est un petit test de calcul d'age de bob et rosita
<p> Année naissance Bob :</p>
<p id="anneeBob">
<p> Année naissance Rosita :</p>
<p id="anneeRosita">
<p Age Moyen :>
<p id="agemoyen">
<script src= "./js/testfctreturn.js"></script>
</body>
</html>
```

testfctreturn.js

```
/* testfctreturn.js petit test de fonction */
function calcMoyAge(annee1, annee2){
      var ageM ;
      var year = new Date().getFullYear() ;
      ageM = (year-annee2+year-annee1)/2 ;
      return ageM ;
}

var moy ;
var an1 = 1980 ;
var an2 = 2000 ;

document.getElementById("anneeBob").innerHTML = an1;
document.getElementById("anneeRosita").innerHTML = an2;
document.getElementById("agemoyen").innerHTML = "Moy="+calcMoyAge(an1,an2);
```

La portée des variables

- Les variables déclarées dans une fonction sont locales à la fonction
- Les variables déclarées dans le script sont visibles dans tout le script.

- Attention aux écrasement liés aux doubles déclarations : globales et locales

Les Events

AVANT :Les écouteurs d'Event

- Exemples : Fin de chargement de page, saisie d'un champ, action sur un bouton, le clavier, la souris

http://www.w3schools.com/jsref/dom_obj_event.asp

Plusieurs type d'Event

- *mouse* (souris):
  - **click** : au clic sur un élément
  - **mouseenter** : la souris passe par dessus la zone qu'occupe un élément
  - **mouseleave** : la souris sort de cette zone
- *keyboard* (clavier)
  - **keydown** : une touche du clavier est enfoncée
  - **keyup** : une touche a été relachée
- window (*fenêtre*)
  - **scroll** : défilement de la fenêtre
  - **resize** : redimensionnement de la fenêtre
- *form* (formulaires)
  - **change** : pour les éléments <input>, <select> et <textarea>, quand l'utilisateur change une de leurs valeurs
  - **submit** : à l'envoi d'un formulaire
- *document*
  - **DOMContentLoaded** : lancé quand le document HTML est complètement chargé et analysé sans attendre que les images et les CSS soient chargés
  - load The browser has finished loading the page

gestion des event

Avant on mettait directement l'evt et l'appel de la fct dans le .html

<button ~~onclick="displayDate()~~">The time is?</button>

Maintenant on utilise les EventListener

Ajout EventListener Exemple avec un <button>

```
<!-- ----- index.html --------- -->
<button id=”butId”>Resize </button>

//---------- main.js ------------------
// recup des id
var but= document.getElementById( "butId");

// Association des événements avec addEventListener
but.addEventListener("click", resize);

// def des fct
function resize(){
        …
}
```

Fonction avec 1 ou plusieurs param

```
// def des fct
function calc(a, b){
        …
}

// recup des id
var but= document.getElementById( "butId");

// Association des événements avec addEventListener
but.addEventListener("click", function() {calc(5,6);});
```

TP

- Faire une page contenant une led et 1 bouton
  - On/Off
- Quand on clique sur le bouton cela allume/éteint
- Le text du bouton change en fct de l'état
- Quand on entre dans le bouton « On/off », le texte devient rouge (/noir)
- Quand on clique la Led et le bouton font un tour

```
// css
        transition: all 1s ease-out ;
// js
                but_on.style.transform="rotate(1turn)" ;
                but_on.style.transform="rotate(-1turn)" ;
```

- Ajouter une video de fin qui apparait quand on quitte la fenêtre

Les Timers

```
// Execution de la fonction maFct après 1s
setTimeout(maFct, 1000);

// Execution de maFct TOUTES les 1s
var ti = setInterval(maFct, 1000);

// arret du Timer
clearInterval(ti);

// Remarque : appel fct avec 1 (ou plusieurs) paramêtres
var ti = setInterval(function() {maFct(param);}, 1000)
```

Petit TP

- Disposer 8 leds côte à côte
- Les faires toutes allumer/éteindre à partir d'un button Start
- Faire l'animation de K2000

La validation de formulaire

EventListener : vérif d'un <input>

```
<!-- .html --------- -->
<form>
<input type="text" id="mon_input" />
</form>

// .js -------------------
// recup des id
var input = document.getElementById("mon_input");
// Association des événements avec addEventListener
input.addEventListener("change", mon_action);

// def des fct
var mon_action = function(){
        //verif des param
}
```

Ajout de paramêtres à une fonction

```
// déclaration fct plusieurs param
function myFunction(p1,p2) {
…
}

// ajout du listener avec function() {…}
element.addEventListener("click", function(){ myFunction(p1, p2); });
```

Exemple de formulaire

```
<form id="formulaire">
        <input type="text" id="nom" placeholder="Nom">
        …
        <input type ="submit" id = "idSubmit" value="Ok"></input>
</form>
<p id="msg"></p>
```

Exemple ‘change’ de formulaire

```
var nomF = document.getElementById( "nom") ;

nomF.addEventListener( "change", validName


function validName () {
        if(nameInput.value.length<1)
                nameInput.style.backgroundColor = "red";
        else
                nameInput.style.backgroundColor = "green";
}
```

Exemple ‘submit’ de formulaire

```
var form = document.getElementById("formulaire") ; // bien prendre le formulaire
var nom = document.getElementsByName("nom")[0] ;
var msg = document.getElementById("msg") ;

form.addEventListener("submit", function (e) {
        var acces = true ;
        console.log( "Nom:"+nom.value) ;

        if (nom.length ===0)
                acces=false ;
        if (!acces)pre
                msg.innerHTML='Entrée interdite';
        else
                msg.innerHTML="Bienvenue " + nom.value ;
        e.preventDefault(); // Annulation de l'envoi des données
});
```

Passage de paramètre (e)

```
// ajout du listener sur le submit
form.addEventListener("submit", function (e) { validForm(e);}) ;

/*
* Validation du formulaire
*/
function validForm(e) {
        console.log("Validation") ;


        e.preventDefault() ;
}
```

Exo, validation formulaire

- Faire un formulaire d'entrée au casino de « Marseille »
- On rentre son nom, son prénom, son âge et son email
- Si l'age est < 18 changer la couleur de la case en rouge
- Affichage d'un message en fonction
  - Interdiction si age < 18
  - Interdiction si on est dans la liste des indésirables : « bob Sinclar», « Gros Raymond », « Joëlle la gazelle »
  - Sinon bienvenue « nom prénom »

Si on veut supprimer un écouteur

```
lien.removeEventListener("click", action_clic);
```

Si on veut récupérer la source d'un évènement

```
// ajout du listener sur le submit
form.addEventListener( "click", function (e)
{ doSomething(e);}) ;

/*
* Validation du formulaire
*/
function doSomething (e) {
        console.log("La source de l'evt" + e.target) ;

}
```

Le DOM

Le DOM

- Document Object Model
  - Référence W3school

Exemple

- window
  - html
    - document
      - head
      - body
        - div
        - a
        - p

Le DOM

- Chaque éléments du DOM est appelé Node

- Quatre types de Nodes
  - Element, ex : <a>
  - Texte, ex: monLien
  - Commentaire, ex: <!-- Et Voilà -->
  - Attribut monLien

Fonction d'intéraction

- document.getElementById(id)
  ```
  <p id="monId" …
  ```
- document.getElementsByClassName(className)
  ```
  <img class="imgSmall" …
  ```
- document.getElementsByName(name)
  ```
  <input type="radio" name="couleur" value="rouge"…
  ```
- document.getElementsByTagName(tagName)
  ```
  <h1 …
  ```
- element.children
- element.childNodes

Propriétés et méthodes des éléments

- innerHTML : non standard (mais implémenté partout), code HTML interne de l'élément
- textContent : contenu textuel de l'élément (tout le texte, en ignorant les balises). Pour IE: à partir v9
- parentNode : nœud parent
- childNodes : tableaux des noeuds fils. Chaque noeud fils est, soit un noeud élément (de nodeType 1), soit un noeud texte (nodeType 3)
- firstChild : équivalent de childNodes[0]
- lastChild : dernier fils
- nodeName : nom de la balise
- nodeType : 1 pour éléments (balises)
- id : identifiant du nœud
- className : classe (au sens CSS) de l'élément
- style : accès aux propriétés CSS de l'élément
- getAttribute(NOM) : valeur d'un de ses attributs. Ne fonctionne pas pour l'attribut "class » sous IE.
- setAttribute(NOM,VALEUR) : pour un noeud/élément, permet de fixer la valeur d'un attribut. Ne fonctionne pas sous IE pour les styles css ni pour l'attribut "class"

Modification des attributs

```
<a href="#top" id="link" class="ma_classe">Top</a>
```

```
mon_lien = document.getElementById("link");

mon_lien.href =  "www.youpi.fr"; // Modifie /le lien vers lequel pointe le <a>
mon_lien.textContent =  "Youpi…" ;

// chgt class (CSS) pour modifier l'affichage
mon_lien.className = "big-blue" ;
```

La création dynamique d'éléments

Création dynamique d'un élément

- Création
  - newElt = document.createElement(tagname)
- Ajout dans le html
  - pere.appendChild(newElt)
- Supression
  - pere.removeChild(newElt);

Exemple

```
var nEl = document.createElement('p');
nEl.textContent="Youpla";
document.body.appendChild(nEl);
```

TP (1/2) 1 Liste d'images

- A partir d'un tableau déclaré dans le .js et contenant une liste de noms, créer automatique la liste des noms en <li>. *createElement…, appendChild…*
- Rajouter un tableau contenant les noms des images des chacun. *nameFiles=["Bob.png…*
- Rajouter sous chaque 'li' les images de chaque personne. *createElement…, appendChild…*
- Changer la taille de toutes les élts 'img' à 40px. CSS ou *getElementsByTagName…*

TP suite : 2 listes d'images

- Ajouter la classe 'imgC' aux images des 'li' (images précédentes). *elt.className=…*
- Entourer toutes les images de la page d'un tour de 2px. Css ou *getElementsByClassName…,*

Suite TP

- Dans la partie droite de la liste
  - Afficher une liste d'objets à vendre avec leur prix et un bouton (+) et un bouton (-) pour ajouter dans le panier
- Dans la partie basse (ou à droite)
  - Afficher les éléments du panier sous forme d'images
  - Afficher la somme
  - Ajouter le bouton pour delete tous les éléments

Les Objets

Les Objets

- Les objets du navigateur
  - window
- Les objets hors-norme (W3C)
  - console

Définir ses propres objets

- Un objet = propriétés + méthodes
- Création d'1 objet

```
var personne = {
     prenom:"Laurent",
     nom:"Freund",
     age : 12,
     fullName : function() {
          return this.prenom + " " + this.nom ;
     }
} ;
```

Accès aux champs, méthode

personne.nom ou personne["nom »]
personne.fullName()

Rem : l'appel sans les parenthèses retourne la définition de la fonction

Exemple :

```
console.debug("age : "+personne.age);
console.debug("nom complet" : "+personne.fullName());

document.getElementById("txt").innerHTML = personne.fullName();
```

Définition d'une classe

- Une classe est un modèle, une définition, une représentation
- function : pour définir la classe
- On utilise une Majuscule comme nom de classe
- this : pour définir des paramètres ou fct accessibles extérieurement

Définition d'une classe

```
function Personne(forname, name, birthYear) {
	this.name = name ; // name accessible ext
	this.forname = forname ; // forname accessible ext

	youngerTen() ; // appel fct interne

	this.age = function() {// age() accessible ext
		var year = new Date().getFullYear() ;
		return year - birthYear ;
	}

	function youngerTen() {// fct interne à la classe
		birthYear += 10 ;
	}
}
```

Définition d'une classe ECMAScript 2015

```
class Personne {
      constructor(name, age) {
            this.name = name ;
            this.age = age ;
      }

      meth1 () {…}
      meth2 () {...}
}

Personne bob = new Personne("Robert", 60);
console.log("The winner is : "+bob.name);
```

Utilisation d'une classe

```
var dir1 = new Personne("Laurent", "Freund", 2000) ; // creation 1er objet
var dir2 = new Personne("Joelle", "Devaux", 2001) ; // creation 2nd objet

document.getElementById("fullnameP").innerHTML = dir1.forname + " " + dir1.name ;
document.getElementById("ageP").innerHTML = dir1.age(); // appel fct

document.getElementById("fullnameM").innerHTML = dir2.forname + " " + dir2.name ;
document.getElementById("ageM").innerHTML = dir2.age(); // appel fct
```

Affichage propriété d'une classe

```
var pers = new Personne("Laurent", "Freund", 2000) ; // creation objet

for (prop in pers)
        console.debug(prop) ; // Affichage de toutes les propriétés de l'objet
```

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Classes

Quelques classes utiles

- Les Timers
- Les String
- Les Nombres
- Les Maths…

Les String

```
var name = "bob";
var i ;
i = str.length ;
i = str.indexof("o");
i = str.lastindexof("b");
i = str.search("b");
i=str.slice(iMin,iMax); // index <0 : à partir de la fin
str.replace("txt1", "txt2");
toUpperCase(), toLowerCase()
str.split(‘,’) ; // conversion tableau: à partir délimiteur
Pour les caractères ",’,\ il faut utiliser \
```

Les Nombres

Entiers : 15 digits
```
var x=9999999999999999 ; //10000000000000000
```

Prbs d'arrondis
```
var x = 0.2 + 0.1;          // 0.30000000000000004
```

Changements de base
```
x.toString(2) ; // en binaire
```

Infinity/-Infinity
```
var x = 2/0;          // Infinity
```

Quelques méthodes des Nombres

toString(n) // Number -> String
toExponantial
toFixed // arrondi
toPrecision
valueOf
Number
parseInt(str) // String-> Number
parseFloat
Number.MAX_VALUE, Number.MIN_VALUE

Math, Date, Array

- http://www.w3schools.com/jsref/jsref_obj_math.asp
- http://www.w3schools.com/jsref/jsref_obj_date.asp
- http://www.w3schools.com/jsref/jsref_obj_array.asp

Debuggage

- Dans le javascript
  - debugger ; // point d'arrêt
- Lancer la console
- Fonctionnalités
  - Pas à pas
  - Cont
  - Points arrets
  - Step into / over / out
  - Inspection variables

Ressources

- **La conférence (slideshow)** : **http://slides.com/liorchamla/javascript#/**
- **Le cours "Javascript pour les débutants" par Le Wagon** (je vous le conseille vraiment !) : **https://www.youtube.com/watch?v=cQZOfeKrWDs** [environ 1h30, c'est de cette conférence que j'ai tiré la progression pédagogique et certains exemples très bien trouvés que je vous ai exposé ce matin et cet après-midi]
- **La formation tout en vidéos et super pédagogique par Jonathan** (site **Grafikart.fr** que je vous conseille désormais de suivre sur YouTube systématiquement) : **https://www.grafikart.fr/formations/debuter-javascript** [environ 8h20 si vous regardez tout]
- **La chaine d'Anthony Welc** qui vous montre tout ce qu'il apprend (il est comme vous ! Il apprend !) : **https://www.youtube.com/channel/UChhPkjgG1-iLUOmURGdgQrw**
- **La formation d'Anthony Welc spécifique sur le Javascript** : **https://www.youtube.com/watch?v=3p8C5jW-7cw&list=PLHSUbP5y6J0UKlYjMkWvU91KwF0Q-th9H&index=2** [un peu plus d'1h]
- **La documentation MDN** : **https://developer.mozilla.org/fr/docs/Web/JavaScript**
- **FreeCodeCamp** : **http://www.freecodecamp.com**
- **La formation Javascript sur CodeCademy** : **https://www.codecademy.com/learn/learn-javascript**

- **Excellent en français**
- http://www.coursweb.ch/javascript/
- codecombat.com