JavaScript

Exercice 1 : "fausse" page web

- créer 1 page web HTML 5 avec 1 titre, des sections, etc.
- ajouter 2 types d'erreurs intentionnelles dans l'ordre proposé (4 erreurs au total pour le validateur)
    - ... ...
    - <table>blabla</table>
- placer une feuille de style avec une bordure pour le tableau.

1. ajouter un script « *inline* » local a une des balises de la page (attribut onload ou onclick par exemple)
2. ajouter un script dans head (balise <script>)
3. ajouter un script à la fin de body (balise <script>)
4. ajouter un script externe (.js) (dans head également) (balise <script>)

Dans un premier temps les 4 codes JavaScript doivent modifier un élément de style (ex. color / backgroundColor)

```
syntaxe 1 : document.getElementById("id").style.color = "red";
```

```
syntaxe 2 : window.onload = function () { ... } ;
```

Exercice 2 : fausse page web corrigée (valide)

- Ajouter une fonction avec paramètre et un appel de fonction dans une des 4 situations précédentes.
- Modifier le contenu (innerHTML) d'un des éléments de la page.
- Vérifier le code source de la page et l'absence d'impact de la modification précédente...
- Vérifier via l'inspecteur DOM que la modification (résultat visible à l'écran dans le navigateur) est bien accessible.

Exercice 3 : gestionnaire d'événement

NB : à partir de cet exercice, l'ensemble du code JavaScript sera placé dans le **fichier externe** (.js) afin d'assurer la séparation des langages.

- Mettre en place un composant de type bouton dans la page HTML
- Mettre en place un écouteur d'événement de type clic sur ce composant qui modifiera le libellé du bouton.

Exercice 4 : vérification d'un formulaire

Concevoir un formulaire HTML 5 contenant 4 ou 5 champs de natures différentes :

un nombre de passager, une adresse mail, une date de départ, une gare de départ (depuis une liste), une gare d'arrivée (depuis une liste également mais dans laquelle la gare de départ sera supprimée ou grisée) avec différents niveaux de vérification :

1. les nouveau type HTML 5 (ex. date)
2. les nouveau attributs HTML 5 (ex. required)
3. les vérification via JavaScript
4. les méthodes GET puis POST
5. le traitement côté PHP avec de nouveau les vérifications utiles.

Suite Exercices JavaScript

Exercice 5 : utilisation des tableaux

- pour générer la liste des gares, il est possible d'utiliser un tableau JavaScritp pour la construction (et les modifications ultérieures) : suggestion.

Exercice 6 : navigation

- simuler via 2 bouton l'équivalent des boutons "précédent" et "suivant" du navigateur (history.back() et history.forward())

Exercice 7 : timer

- affichage d'un compte à rebourd de 1 mn et à 0 le formulaire devient grisé : pour afficher le temps restant, il faut utiliser setInterval()

Exercice 8 : mise en évidence d'éléments

- créer quelques boutons sur votre page permettant sur un événement de type "clic" de mettre en évidence (par exemple avec une couleur de fond) tous les paragraphes / tous les titres de la page (le bouton changera d'état et un nouveau clic permet d'enlever la mise en évidence).
- faire de même avec une classe CSS spécifique.

Exercice 9 : exploration de l'objet window

créer une fonction recursive (limité aux 3 premiers niveaux) qui affiche sous la forme de listes imbriquées les attributs de l'objet window.

Exercice 10 : API Google Map

- mettre en oeuvre l'API google Map en centrant la carte (par exemple zoom 10) sur le bâtiment de St Martin
- ajouter une punaise sur la position.
- conserver ce comportement par défaut et ajouter la géolocalisation.
- pour la clef, il faut posséder un compte gmail : cf. https://developers.google.com/maps/faq?csw=1#using-google-maps-apis pour la procédure.

1. Quelques conseils pour le débogage JavaScript

- utiliser Chrome ou Firefox
- utiliser l'outil console de votre navigateur (Ctrl + Shift + K sous Firefox)
- utiliser l'inspecteur DOM (intégré au "web tools" du navigateur ou via un plugin)
- utiliser les méthodes de l'objet "console" de JavaScript :
    - `ex. console.debug("...");`

•

Compléments JavaScript

- principales méthodes : getElementById, getElementsByName, getElementsByTagName, getElementsByClassName : attention à la casse et au pluriel (sauf pour getElementById)
- manipulation du DOM (utiliser DOM inspector ou équivalent) :
    - document >> HTML >> BODY >> DIV >> H1 >> text
    - quelques exemples de manipulation : innerHtml, setAttribute(), removeAttribute()
- tableaux :
    - var tab = ["...", "...", "..."];
    - var tab = new array("...", "...", "...");
    - => dans ces 2 exemple le parcours se fait par : for(var i = 0 ; i < tab.length ; i++) { ... }
    - var tab2 = { clef1 : "..." , clef2 : "..." } // dans ce cas le tableau est initialisé comme un objet et les clefs correspondent aux attributs.
- méthodes GET et POST pour le transfert du formulaire depuis le navigateur vers le serveur web.
- certains éléments sont nativement disponibles : exemple :

```
document.forms correspond à document.getElementsByTagName("form")
frm1.elements
exemple de syntaxe :
for (var i = 0 ; i < document.getElementsByTagName("form")
[0].elements.length ; i++) {
   document.forms[0].elements[i].disabled = true;
}
```