**JAVASCRIPT**

**I Rappels sur Javascript**

Le véritable nom de javascrip est ECMAscript dasn le cadre de la spécification ECMA-262.

Javascript est un langage développé par Netscape en 1995, dérivé d’un projet de Brendan Eich, et ajouté à Netscape Navigator, puis par la suite aux autres navigateurs..

Ce langage interprété par l’agent utilisateur (navigateur) doit permettre d’ajouter de l’intéraction avec l’utilisateur au lieu d’un simple chargement de page statique.

Il doit permettre aussi au travers du DOM (Document Object Model) de manipuler les éléments HTML présents dans la page : les créer dynamiquement, les modifier, les supprimer, agir sur leurs propriétés CSS, etc..

C’est aussi Javascriot qui est mis en oeuvre au sein de la technologie Ajax (Asynchronous Javascrip and XML) pour effectuer des chargements de contenu dynamique en relation avec le serveur, ou encore JSON(Javascript Object Notation) en tant que format de données.

**II- Frameworks Javascrip**

Un « framework » Javascript est un ensemble de fonctions (écrites elles-même en javascript), regroupées dans un fichier externe à inclure dans une page HTML grâce à la balise <script>. Un framework définit un environnement permettant d’étendre les possibilités déjà offertes par la base du langage.

JQuery (Licence GPL/MIT) est un framework très connu. Il comprend des fonctions de modification dynamique du DOM, des styles CSS, des effets d’animation, et des fonctions pour l’écriture de gestionnaires d’évènements ou d’appels Ajax.

Il y a d’autres frameworks comme : MooTools, Dojo Toolkit, Scriptaculous

**III- Le placement du script**

<head>

<**script src= "test.js"**></script>

...........

</head>

OU avant <**/body**> ou dans le contenu de la page.

**IV- Les bases de Javascript**

Dans la majorité des cas, on utilise Javascript dans une page Web pour manipuler le DOM.

C’est quoi le DOM ?

Le DOM est une interface de programmation pour tous les documents et pages Web ou XML. Il s’agit d’une structure représentant ces documents tels qu’ils ont été interprétés en mémoire, permettant de parcourir et modifier leur contenu, ainsi que leur présentation visuelle si des règles CSS sont appliquées.

Au chargement d’un document HTML, un arbre de la page est élaboré via le DOM. Chacun des éléments est considéré comme un objet et peut jouer le rôle de noeud, auquel on peut attacher les qualités de parent, enfant,orphelin ou frère. Cette représentation permet d’accéder aux propriétés de chacun de ces objets ou groupe d’objets et aux méthodes qui les équipent.

Les opérations les plus courantes sont l’ajout, la modification et la suppression de noeud.

**Exemple** l’arborescence DOM d’un document HTML est comme suit :

<html>

<head>

<body>

<script>

<header>

<section>

<article>

<footer>

<h1>

<p>

<p>

Texte

Texte

<audio>

*1- Variables*

Comme tout langage de programmation, on retrouve de nombreuses structures connues et la notion de variable.

Avec HTML, on retrouve la déclaration de script dans une balise <script>

<script>

alert("Script") ;

</script>

Toute instruction est séparée par « ; ».

Une variable est déclarée par le mot-clé « **var** » suivi de « **=**»

**Exemple**

var a=3 ;

var b=4 ;

alert(a\*b) ;

*Remarque*

Pour profiter au mieux de la découverte de javascipt et surtout de meilleures conditions de développement et débogage, utiliser un navigateur moderne équipé d’une console Javascript et utiliser plutôt la fonction console.log().

**2- Types simples**

Des chaînes de caractères peuvent être déclarées :

var proverbe= "rien ne sert de courir il faut partir à point" ;

et concaténées grâce au signe «**+** »

console.log(proverbe+ "claire fontaine") ;

Des tableaux de données sont initialisés rapidement :

var tableau=Array("Pommes ","ananas", "orange") ;

Chaque élément de tableau est accessible par son index (commence par zero) placé entre crochets «**[]** ».

console.log(tableau[0]) ; //*Affiche Pommes*

console.log(tableau[2]) ; //*Affiche orange*

La longueur d’une chaîne de texte ou d’un tableau peut être connue grâce à sa propriété length.

proverbe.lenght ;//**45**

tableau.lenght ; //**3**

tableau[2].lenght ; //**6**

**3- Objets**

Les objets sont crées avec des paires de clés/valeurs, séparées par des virgules, le tout regroupé entre accolades « { } »

**Exemple**

var voiture={

roues :4,

couleur :"rouge",

achat :{

prix :20000,

année :2011,

}

} ;

Un objet peut contenir un sous-objet, et ainsi de suite. Dans notre cas l’objet « **voiture**» contient trois propriétés (**roues,couleur et achat**) dont la dernière est un autre objet avec deux propriétés (**prix, année**).

Les propriétés des objets sont accessibles en « naviguant » dans l’arborescence de l’objet par des points « . »

voiture.achat.année ; //*Affiche 2011*

Un appel à console.log permet d’obtenir une visualisation de la variable objet sous forme d’arbre.

console.log(voiture) ;

Certains objets sont célèbres dans un navigateur, ils servent d’interface à un ensemble d’informations réunies au chargement du document. Parmi eux, window qui « symbolise » la fenêtre active.

window ; //*interface DOMWindow*

window.document ; //*interface Document*

window.navigator ; // *interface Navigator*

window.navigator.userAgent ; /*/de type String*

En programmation orientée objet, l’objet lui-même est une instance de classe.

var o=new Object() ;

Le mot clé « new » intervient, suivi de l’appel au constructeur (le nom de la classe de l’objet).

var w=new Worker() ;

Il s’agit d’un objet implémentant l’interface Worker.

Tout est objet en Javascript, avec l’exception de « null » et « undefined » qui correspondent respectivement à une valeur nulle(différente de zero), et une valeur indéfinie.

var s=new String("Rose") ;

s.toString() ; // *Affiche "Rose"*

**4- Fonctions**

Les fonctions servent à regrouper des instructions. Elles sont déclarées avec le mot-clé « **function** » suivi du nom de la fonction et «  **( )** ».

**Exemple**

function ident\_navigateur(){

console.log(window.navigator.userAgent) ;}

cette function est invoquée comme suit :

ident\_navigateur() ; // *Affiche le navigateur*

**Une variante**

ident\_navig=function() {

console.log(window.navigator.userAgent) ;}

Un autre exemple de fonction avec paramètres

function addition(a,b){

return a+b ;

var c=addition(2,3) ;

console.log(c) ;

**5- Boucles**

- **For**

for(var i=0 ;i<10 ;i++){

console.log(i) ; // Affiche 0,1,2,....9}

- **while**

var i=0 ;

while(i<10){

console.log(i) ;

i++ ;

}

**V- Méthodes de sélection DOM**

Les fonctions pour la selection de noeuds dans le DOM sont getElementById et getElementsByTagName(depuis DOM Level 2). Elles permettent de cibler un élément par son identifiant, l’attribut id, ou par son nom. Une fois l’objet obtenu par retour de ces fonctions, on peut accéder en lecture ou en modification à ses propriétés (attributs, style,...).

getElmentById()

Renvoie un élément dont on spécifie l’identifiant (attribut "id") ou null sinon

**Exemple**

<h1 id= "titre">HTML5</h1>

<script>

var element=document.getElementById("titre") ;

alert(elment.textcontent) ; //*Affiche "HTML"*

</script>

getElmentsByTagName()

Renvoie un ensemble d’éléments dont le type est spécifié.

var elements=document.getElmentsByTagName("img") ;

La variable elements obtenue est de type nodeList. Chacun de ses éléments est accessible de deux manières :

elements.item(1) ;

elements[1] ;

D’autres fonctions ont été crées

getElementsByClassName()

Renvoie un ensemble d’éléments dont on spécifie la classe.

var elements=document.getElementsByClassName("maclasse ") ;

querySelector() (Nouveau HTML 5)

Retourne le premier élément qui correspond au sélecteur spécifié(CSS)

var el=document.querySelector("#menu ul li.sousmenu") ;

querySelectorAll() (HTML 5)

Retourne tous les éléments qui correspondent au sélecteur spécifié.

var titres=document.querySelectorAll(" article>h1") ;

**VI- Propriétés et Méthodes DOM**

Les éléments DOM renvoyés par les fonctions précédentes sont de type HTMLElement lorqu’on a un élément, et HTMLDOMChildrenCollection lorqu’on obtient une énumération d’un ou plusieurs éléments.

Quelques propriétés ou méthodes de l’interface HTMLElement

| **Propriétés ou Méthodes** | **Description** | **Type** |
| --- | --- | --- |
| innerHTML | Contenu HTML d’un élément | string |
| outerHTML | Contenu HTML et l’élément lui-même | string |
| textContent ou innerText | Contenu Texte simple | string |
| style | Ensemble des propriétés CSS de l’élément | CSSStyleDeclaration |
| style.\* | Propriétés CSS individuelles, accessibles par des membres :  style.width, style.fontSize, style.listStyleType | string |
| tabindex | Index de tabulation | long |
| id | Valeur de l’attribut « **id** » | string |
| lang | Valeur de l’attribut « **lang** » | string |
| title | Valeur de l’attribut « **title** » | string |
| tagName | Type de l’élément (nom de la balise) | string |
| ClassName | Valeur de l’attribut de classe affecté à l’élément | string |
| classList (HTML 5) | Enumération des classes individuelles présentes dans l’attribut « class » | DOMTokenList |
| childrenNodes | Liste des nodes (noeuds) enfants contenus dans l’élément | NodeList |
| firstChild | Premier Node enfant | HTMLElement |
| lastChild | Dernier Node enfant | HTMLElement |
| parentNode | Node parent | HTMLElement |
| nextSibling | Node frère suivant | HTMLElement |
| prevSibling | Node frère précédent | HTMLElement |
| attributes | Attributs de l’élément | NamedNodeMap |
| getAttribute(attr) | Renvoie la valeur de l’attribut «**attr**» |  |
| setAttribute(attr,val) | Modifie la valeur de l’attribut « **attr** » par **val** |  |
| removeAttribute(attr) | Supprime l’attribut « **attr** » s’il existe |  |
| focus() | Donne le focus à l’élément |  |
| blur() | Retire le focus. |  |

**VII-Manipulation DOM**

createElement()

Crée dynamiquement un noeud DOM. Le nom de l’élément HTML est indiqué (sans chevrons).

var nouveauDiv=document.createElement("div") ;

nouveauDiv.innerHTML= "<p> C’est un <b>nouveau </b> paragraphe </p>" ;

Cet élément est crée mais il devient visible qu’après insertion à l’aide d’une des fonctions suivantes

appendChild()

Insère un noeud DOM enfant dans un parent

var paragraphe=document.createElement("p") ;

paragraphe.innerHTML= "je suis<b>un nouveau</b>" ;

document.body.appendChild(paragrahpe) ;

removeChild()

Retire un enfant à son parent.

var paragraphe=document.querySelector(" body>p") ;

document.body.removeChild(paragraphe) ;

insertBefore()

Insère un noeud DOM avant un autre.

var nouveauLi=document.createElement("li") ;

nouveauLi.innerHTML= "je suis <b> nouveau </b> ";

var uneListe=document.getElementByTagName("ul")[0] ;

uneListe.insertBefore(nouveauLi,uneListe.lastchild) ;

createTextNode()

Crée un noeud Texte

var nouveauTexte=document.createTextNode("beau temps") ;

document.querySelector("body").appendChild(nouveauTexte) ;

classList (HTML 5)

cette propriété manipule l’attribut « **class** » avec des méthodes appropriées qui manquaient auparavant et devaient être comblées par des bibliothèques Javascript supplémentaires.

var monElment=document.getElemmentById("résultata") ;

monElement.classList.add("positif") ;

**A- Les propriétés et méthodes de classList**

| **Propriétés ou Méthodes** | **Description** | **Retour** |
| --- | --- | --- |
| length | Nombre de classes appliquées à l’élément | integer |
| add(classe) | Ajoute une classe à l’ensemble |  |
| remove(classe) | Supprime une classe |  |
| toggle(classe) | Ajoute ou supprime une classe, alternativement, selon son état initial |  |
| contains(classe) | Retourne true en cas de présence d’une classe, ou false sinon | boolean |
| toString() | Retourne une chaîne de texte contenant toutes les classes appliquées | string |

insertAdjacentHTML() (HTML 5)

Cette méthode insert du contenu dans un document HTML. Elle prend comme premier paramètre l’emplacement d’insertion :beforebegin (avant l’élément), afterbegin (dans l’élément au début, avant son premier enfant), beforeend (à la fin de l’élément, après son dernier élément), afterend (après l’élément), et en deuxième paramètre le bout de code HTML à ajouter.

**Exemple**

<ul id= "maliste ">

<li> Premier </li>

<li> Deuxième </li>

</ul>

<script>

var uneListe=document.getElementById("maliste") ;

uneListe.insertAdjacentrHTML("beforeend", "<li> je suis le dernier </li> ") ;

**VIII- Méthodes pour Formulaires**

Il existe des méthodeq propres aux API d’éléments particuliers. Parmi eux , les champs d’entrée de type <**inpu**t>

**VIII-1 Quelques propriéts et méthodes**

| **Propriétés et Méthodes** | **Description** | **Retour** |
| --- | --- | --- |
| value | Retourne la valeur du champ  Peut être modifié (avec=) pour changer la valeur. |  |
| checked | Retourne true si coché, ou false si décoché | boolean |
| files | Liste des fichiers sélectionnés. Renvoie null si ce n’est pas un champs d’entrée de fichiers. | FileList |
| valueAsDate | Peut être modifié (avec=) pour changer la valeur | Date |
| valueAsNumber | Peut être modifié (avec=) pour changer de valeur | Number |
| list | Retourne l’élément <**datalist**> indiqué par l’attribut « **list**» | HTMLElement |
| selectedOption | Retourne l’option de liste sélectionnée | HTMLElement |
| stepUp(n) | Incrémente la valeur de n |  |
| stepDown(n) | Décrémente la valeur de n |  |

**IX- Gestionnaire d’évènements**

Ces gestionnaires sont des fonctions Javascript associées à un évènement utilisateur ou navigateur. Une fois l’évènement déclenché (par exemple un clic, une touche pressée, ou une page chargée), la fonction est exécutée.

**IX-1 Les attributs évènements**

Les attributs gestionnaires jouent un rôle particulier car ils associent un type d’évènement DOM pouvant survenir pour un élément à une fonction de script.

Cela permet de déclencher un bout de code Javascript sur des évènements tels que « lorsque la page a été chargée », ou « lorsque l’utilisateur presse sur une touche ». La plupart d’entre eux concernent :

- les entrées utilisateur (souris,clavier) ;

- la navigation au clavier (focus) ;

- les formulaires (changement de valeur, validation) ;

- la lecture de médias (audio,vidéo) ;

- les intéractions riches (glisser-déposer, molette souris) ;

- le chargement des pages et des ressources.

**Déclenchement d’une fonction au clic**

<input type= "button" value="Cliquez-ici" onclick= "faireQuelquechose() ; ">

<script>

function faireQuelquechose(){

.........

alert("Un clic ") ;

}

</script>

Certains sont nouveaux en HTML 5, surtout ceux qui concernent la lecture des médias (audio,vidéo). Tous débutent par le terme « **on** » suivi du nom de l’évènement DOM auquel ils correspondent.

**IX-2 Les évènements DOM sont :**

| **Evènement** | **Correspondance** | **Catégorie** |
| --- | --- | --- |
| onabort | Chargement annulé de l’utilisateur | Réseau |
| onblur | Perte de focus : l’élément détenait le focus et l’a perdu, car l’utilisateur a navigué vers un autre élément, au clavier ou grâce à son dispositif de pointage | Navigation |
| oncanplay | L’agent utilisateur peut reprendre de l’audio ou la vidéo, mais estime que s’il débute à cet instant, la totalité du média ne pourra être lue au rythme actuel sans avoir à effectuer d’autre(s) pause(s) pour mettre en cache les données | Médias |
| oncanplaythrough | L’agent utilisateur peut reprendre la lecture de l’audio ou de la vidéo en estimant qu’à cet instant la totalité du média pourra être lue au rythme de lecture actuel sans avoir à effectuer d’autre(s) pause(s) pour mettre en cache les données | Médias |
| onchange | L’utilisateur a validé le changement de la valeur d’un élément (de formulaire) | Formulaires |
| onclick | L’utilisateur a cliqué sur l’élément et a relâché le bouton de la souris ou a utilisé un autre dispositif de pointage qui émule cette action | Souris,pointage |
| oncontextmenu | L’utilisateur a déclenché le menu contextuel | Navigation |
| ondblclick | L’utilisateur a double-cliqué sur l’élément ou a utilisé un autre dispositif de pointage qui émule cette action | Souris,pointage |
| ondrag | L’utilisateur est en train de déplacer l’élément | Drag&drop |
| ondragend | L’utilisateur a fini de déplacer l’élément | Drag&drop |
| ondragenter | L’opération de glisser-déposer initiée par l’utilisateur est entrée dans l’élément | Drag&drop |
| ondragleave | L’opération glisser-déposer initiée par l’utilisateur a quitté l’élément | Drag&drop |
| ondragover | L’opération glisser-déposer survole l’élément | Drag&drop |
| ondragstart | Début de glisser-déposer | Drag&drop |
| ondrop | Fin de glisser-déposer | Drag&drop |
| ondurationchange | L’attribut DOM «**duration** » sur l’élément <audio> ou <video> a été modifié | Médias |
| onemptied | L’élément <audio> ou <video> a été modifié | Médias |
| onended | La fin de l’audio ou de la vidéo a été atteinte | Médias |
| onerror | Le chargement de l’élément a échoué | Réseau |
| onfocus | L’élément a reçu le focus | Navigation |
| onformchange | L’utilisateur a validé le changement de valeur d’un élément dans le formulaire dont cet élément dépend | Formulaires |
| onforminput | L’utilisateur a modifié la valeur d’un élément dans le formulaire dont cet élément dépend | Formulaires |
| oninput | L’utilisateur a changé la valeur de l’élément formulaire | Formulaires |
| oninvalid | l’élément de formulaire n’a pas respecté les contraintes de validation | Formulaires |

**Les évènements (suite)**

| **Evènements** | **Correpondance** | **Catégorie** |
| --- | --- | --- |
| onkeydown | L’utilisateur a enfoncé une touche | Clavier |
| onkeypress | L’utilisateur a enfoncé une touche correspondant à un code caractère | Clavier |
| onkeyup | L’utilisateur a relâché une touche | Clavier |
| onload | L’élément a fini son chargement | Réseau |
| onloadeddata | L’agent utilisateur peut rendre le contenu audio ou vidéo à la position de lecture courante, pour la première fois | médias |
| onloadedmetadata | L’agent utilisateur a pu déterminer la durée et les dimensions de l’élément <audio> ou<video> | Médias |
| onloadstart | L’agent utilisateur a commencé à charger des données médias dans l’élément <audio> ou<video> | Médias |
| onmousedown | L’utilisateur a pressé le bouton de la souris ou du dispositif de pointage sur l’élément | Souris,pointage |
| onmousemove | L’utilisateur a effectué un mouvement avec la souris ou le dispositif de pointage | Souris,pointage |
| onmouseout | L’utilisateur a déplacé le curseur de la souris ou du dispositif de pointage hors des limites de l’élément | Souris,pointage |
| onmouseover | L’utilisateur a déplacé le curseur de la souris ou du dispositif de pointage à l’intérieur des limites de l’élément, ou d’un de ses descendants | Souris,pointage |
| onmouseup | L’utilisateur a relâché le bouton de la souris ou du dispositif de pointage sur l’élément | Souris,pointage |
| onmousewheel | L’utilisateur a actionné la molette de défilement de la souris ou du dispositif de pointage équivalent | Souris,pointage |
| onpause | L’utilisateur a mis en pause la lecture de l’élément <audio> ou<video> | Médias |
| onplay | L’agent utilisateur a débuté la lecture de l’élément <audio> ou<vidéo> | Médias |
| onplaying | La lecture de l’élément <audio> ou<video> a débuté | Médias |
| onprogress | L’agent utilisateur télécharge les données médias pour l’élément <audio> ou <vidéo> | Médias |
| onratechange | L’attribut DOM « **defaultPlaybackRate** » ou »**playbackRate** » a été modifié sur l’élément <audio> ou <vidéo> | Médias |
| onreadystatechange | L’élément et toutes ses ressources ont fini leur chargement | Médias |
| onreset | Le formulaire a été remis à zéro | Formulaires |
| onscroll | L’élément qui a fait l’objet d’un défilement | Souris,pointage |

**Les évènements (Suite)**

| **Evènements** | **Correspondance** | **Catégorie** |
| --- | --- | --- |
| onseeked | La valeur de l’attribut « **seeking** » a été modifiée à false (l’opération de navigation temporelle sur l’audio ou la vidéo a pris fin) | Médias |
| onseeking | La valeur de l’attribut «**seeking** » a été modifiée à true (l’opération de navigation temporelle sur l’audio ou la vidéo a eu lieu et est d’une durée suffisamment pertinente pour déclencher l’évènement | Médias |
| onselect | L’utilisateur a sélectionné du texte | Souris,pointage |
| onshow | L’utilisateur a requis l’affichage de l’élément comme menu contextuel | Navigation |
| onstalled | L’agent utilisateur essaie de télécharger les données pour la lecture de l’élément média <audio> ou <video>, mais celles-ci ne sont pas reçues | Médias |
| onsubmit | Le formulaire a été validé | Formulaires |
| onsuspend | L’agent utilisateur ne télécharge actuellemnt pas le contenu du média, mais ne dispose pas encore de toutes les données | Médias |
| ontimeupdate | La position de lecture de l’élément <audio> ou<video> a changé | Médias |
| onvolumechange | L’attribut DOM « **volume**» ou « **muted** » a été modifié sur un élément média | Médias |
| onwaiting | La lecture de l’élément <audio> ou<video> a été mise en pause, car les données pour la suite sont attendues | Médias |