I. Introduction :

Le JavaScript est un langage de scripts principalement utilisé dans les pages web interactives.

Ce langage permet d'apporter des améliorations au langage HTML en permettant d'exécuter des commandes du côté client, c'est-à-dire au niveau du navigateur et non pas du serveur web.

Activité :

Ecrire un programme en HTML permettant de calculer la somme de deux variables a et b.

Constatation :

Nous avons besoin de déclarer deux variables a, b et une troisième variable S qui va contenir la somme.
Ce qui est impossible à réaliser par le langage HTML.

Limites du langage HTML :

- *Absence des structures de contrôles algorithmiques (conditionnelles et itératives).*
- *Aucune logique de programmation procédurale (variables, sous programmes, …).*
- *Code source accessible par les utilisateurs (menu affichage du navigateur --> code source).*
- *…*

II. Structure d'un document JavaScript :

Activité :

*Taper le code ci-dessous puis enregistrer le fichier sous le nom **TP1.html***

```
<HTML>                                          }
<HEAD>                                          }
<TITLE> Structure code JavaScript </TITLE>      } Balises HTML
</HEAD>                                         }
<BODY>                                          }
Texte en HTML  <br><br>                         }

<script language ="JavaScript">                 }
alert ("Bienvenue");                            } Script
// Alerte affiche une fenêtre message.          }
</script>                                       }

Un autre texte en HTML                          }
</BODY>                                         } Balises HTML
</HTML>                                         }
```

Constatations :

- *On utilise la balise **< script language ="JavaScript">** … **</script>** pour signaler au navigateur qu'il s'agit d'un code JavaScript.*
- *Pour ajouter un commentaire sur une seule ligne on utilise : //……*
- *Pour ajouter un commentaire sur plusieurs lignes on utilise : /*……*/*

III. Les objets JavaScript: TP1

1. Notion d'objets :

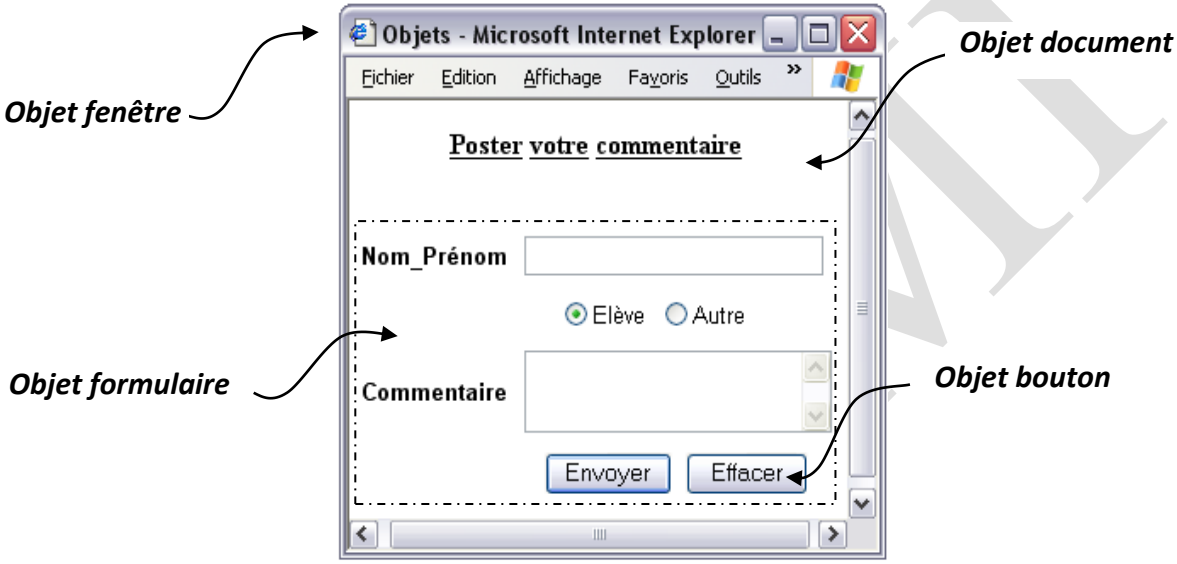
Les objets sont les éléments de base permettant d'accéder aux différentes fonctionnalités offertes par JavaScript.

Il existe deux types d'objets :

- *Les objets d'interface tels que **fenêtre**, **document**, **formulaire**, **radio**, **bouton**, …*
- *Les objets des propriétés et des fonctions prédéfinies tels que **math**, **date**, **array**, …*

2. Hiérarchie des objets d'interface :

Exemple:



La hiérarchie des objets de cet exemple est :

Fenêtre** --> **Document** --> **Formulaire** --> **Zone de saisie** / **Bouton radio** / **Bouton

Pour accéder à un objet, il faut donner le chemin complet de l'objet en commençant du contenant le plus extérieur.

*Soit par exemple pour le bouton radio "Elève" : **(window).document.form.radio[0]***

*L'objet **window** occupe la première place dans la hiérarchie, il est repris par défaut par JavaScript et devient donc facultatif.*

***NB** : radio[0] : désigne le premier bouton radio rencontré dans le formulaire.*

3. Propriétés des objets :

Pour accéder aux propriétés des objets JavaScript, on utilise la syntaxe :

nom_de_l'objet . nom_de_la_propriété

*Par exemple, l'une des propriétés d'un bouton radio est sa sélection ou sa non-sélection (**checked**).*

Dans le cas du bouton radio "Elève", pour tester la propriété de sélection, on écrit :

If (document.form.radio[0].checked)

4. Méthodes des objets:

*Pour chaque objet JavaScript, ils existent plusieurs méthodes tel que : la méthode **write** de l'objet **document**, les méthodes **alert** et **prompt** de l'objet **window**.*

IV. Les variables: TP1

1. Déclaration :

Une variable est un objet défini par son nom et son contenu.
Elle peut être déclarée de deux façons :

- ***Explicitement** : le nom précédé par le mot clé **var** (var x = 3;).*
- ***Implicitement** : le nom suivi par la valeur (x = 3).*

2. Types de données :

En JavaScript, on peut utiliser 4 types de données :
- Les nombres : entiers et réels
- Les chaines de caractères
- Les booléens : true ou false
- Le mot null : indique l'absence d'une valeur

Remarques:

1. *Il existe des fonctions de conversion explicite de type : **String** et **Number***
   Var a = String(13.5) --> a contient la chaine "13.5"
   Var b = Number ("14.2") --> b contient le réel 14.2
2. *Il est préférable, de précéder toute conversion avec la fonction **Number**, par un test de validité avec la fonction **isNaN***

Application : TP1

V. Les opérateurs prédéfinis:

1. Les opérateurs de calcul:

*+, -, *, /*
% modulo (MOD)
= affectation (:=)

2. Les opérateurs de comparaison:

<, <=, >, >=
== égale
!= différent

3. Les opérateurs logiques:

&& représente ET (AND) et || représente OU (OR)

4. Les opérateurs d'incrémentation:

Ces opérateurs augmentent ou diminuent la valeur d'une donnée par 1
i ++; i <-- i+1
i - -; i <-- i-1
Exemple :
x = 2; y = x++; --> y va contenir 3

5. Les opérateurs associatifs:

*+=, -=, /=, *=*
x += y --> x = x + y
*x *= y --> x = x * y*

VI. Les entrées/sorties:

Syntaxes :

- ***L'entrée (saisie):** nom_variable = prompt ("texte","valeur par défaut");*
- ***La sortie (affichage):** document.write ("message" + nom_var);*
  ou Alert ("message" + nom_var);

Application : TP1

```
<script language="JavaScript">
x = Number(window.prompt ("Donner x : ",""));
y = Number(window.prompt ("Donner y : ",""));
s = x+y;
alert("La somme de  "+x+" et  "+y+" est : "+s);
</script>
```

VII. Les structures de contrôle: TP1

1. Les structures conditionnelles:

a. La structure IF:

La forme réduite :

```
if (condition vraie)
{
traitement
} (; non)
```

La forme complète :

```
if (condition vraie)
{
traitement1
}
else
{
traitement2
}
```

b. La structure switch:

```
switch (expression)
{
 case val1 : Traitement 1 ;  break;
 case val2 : Traitement 2 ;  break;
 …
 default :  Traitement n ;  break;
}
```

Remarques:

- L'instruction **break;** permet de quitter la structure switch après l'exécution du bloc.
- La partie **default** ($n^{ème}$ traitement) est facultative.

2. Les structures iteratives:

a. La structure for:

```
for (Initialisation ; condition ; progression)
{
Traitement
}
```

Exemple :

Donner l'ensemble d'instructions permettant d'afficher **Ligne : i** avec i un entier allant de 0 à 9.

```
for (i=0; i<10; i++) {document.write("ligne : " + i + "<br>")}
```

En Pascal:

```
for i:=0 to 9 do writeln(("ligne : ",i);
```

b. Structure do..while:

```
do
{
Traitement
}
while (condition de continuité)
```

Exemple :

```
i=0 ;
do {
   document.write("ligne : " + i+"<br>") ;
   i++ ;
   }
while(i<10)
```

c. Structure While:

```
while (condition)
{
Traitement
}
```

Exemple :

```
i=0;
while (i<10) {
document.write ("ligne : " + i + "<br>");
i++;
}
```

VIII. Les fonctions: TP1

En JavaScript, il existe deux types de fonctions :

- Les fonctions prédéfinies (méthodes) : elles sont associées à des objets bien particuliers, tel que la méthode Alert() avec l'objet window.
- Les fonctions déclarées par l'utilisateur.

1. Déclaration:

La syntaxe de déclaration d'une fonction est la suivante :

```
function nom_fonction (paramètres)
{
traitement
return résultat;
}
```

2. L'appel d'une fonction:

Pour appeler une fonction on écrit : nom_fonction (paramètres);

Remarques :

- On peut définir une fonction dans l'entête de la page : entre <HEAD> et </HEAD>
- L'instruction **return** est facultative, une fonction sans retour est équivalent à une procédure.

IX. Les événements: TP1

Les événements JavaScript, associés aux fonctions, aux méthodes et aux formulaires, offrent la possibilité de création d'une réelle interactivité entre l'utilisateur et les pages Web.

Pour chaque événement on associe l'action prévue en respectant la syntaxe suivante :

< nom_balise OnEvénement = "fonction()">

Avec :

OnEvénement représente un attribut associé à une balise HTML (**nom_balise**).

Fonction() : appel d'une fonction

Dans ce qui suit, on va s'intéresser aux événements JavaScript : ***onclick****,* ***onfocus*** *et* ***onchange****.*

1. onClick :

L'événement OnClick est utilisé avec les boutons et les liens hypertextes.
Il est aussi possible de programmer l'événement OnClick avec les objets : case à cocher, case d'option et zone de liste

2. onFocus *:*

Cet événement survient lorsqu'un champ de saisie est prêt à recevoir ce que l'utilisateur a l'intention de taper au clavier.

3. onChange :

Cet événement survient lorsque la zone de saisie perd le focus après modification du contenu.
Onchange est utilisé avec les balise <textarea> et <select>.

4. onSubmit :

Cet événement, spécifique à la balise form, vérifie le contenu du formulaire avant d'être envoyé.

X. Les formulaires: *TP2*

En JavaScript on peut accéder à chaque élément d'un formulaire pour lire ou saisir une donnée, faire un choix,… ce qui rend les pages Web plus interactives.
Pour déclarer un formulaire en JavaScript on utilise la balise: ***<form.name = "****nom_formulaire****">***

1. Le contrôle zone de texte:

Pour créer une zone texte en JavaScript, on utilise la syntaxe suivante :

<input type = "*text****" name = "****nom****" value = " " size = "x" maxlenght = "y">***

L'objet text possède 3 propriétés :

- ***name*** *: le nom du contrôle*
- ***default value*** *: valeur par défaut affichée dans la zone de saisie.*
- ***value :*** *indique la valeur en cours dans la zone de texte*

On peut affecter la valeur d'une zone de saisie à une variable à l'aide de la syntaxe suivante:

NomVariable = document.NomFormulaire.NomZone.value

Pour modifier la valeur d'une zone text :

document.NomFormulaire.NomZone.value = NomVariable

2. Les boutons radio:

Ce bouton permet de faire un seul choix parmi plusieurs propositions.
L'objet radio possède les propriétés suivantes :

name *: indique le nom du bouton radio, tous les boutons ont le même nom.*
index *: l'index ou le rang du bouton radio en commençant par 0.*
checked : *indique l'état en cours de l'élément radio (sélectionné ou non).*
defaultchecked : *indique l'état du bouton par défaut.*
Value: *indique la valeur de l'élément radio.*

- *Pour vérifier l'état d'un bouton on doit utiliser cette syntaxe :*

NomVariable=document.NomF.NomCase[indice].checked

NomVariable *est une variable booléenne*

- *Pour récupérer la valeur d'un bouton radio:*

NomVariable=document.NomF.NomCase[indice].value

- *Pour connaître le nombre d'options:*

NomVariable=document.NomF.NomCase.length

3. Les boutons case à cocher:

Ce bouton permet de faire un ou plusieurs choix parmi un ensemble de propositions.
La manipulation des boutons cases à cocher est semblable à celle des boutons radio, la seule différence concerne la propriété ***name*** *: toutes les cases à cocher portent des noms différents.*

4. Les listes de sélection:

- *Les propriétés d'une liste de sélection :*

Name *: indique le nom de la liste déroulante.*
length : *indique le nombre d'éléments de la liste.*
S'il est indiqué dans la balise ***<SELECT>****, tous les éléments de la liste seront affichés. Si vous ne l'indiquez pas un seul apparaîtra dans la boite de la liste déroulante.*
selectedIndex *: indique le rang à partir de 0 de l'élément de la liste qui a été sélectionné par l'utilisateur.*
defaultselected *: indique l'élément de la liste sélectionné par défaut. C'est lui qui apparaît alors dans la petite boite.*

- *Les propriétés des éléments d'une liste de sélection :*

selected : *renvoie true si l'option est sélectionnée et false sinon.*
text : *renvoie le texte de l'élément sélectionné*
value : *renvoie la valeur de l'élément sélectionné*

- *Pour ajouter une nouvelle option à la liste:*

```
NomVariable=new Option (texte, valeur)
document.NomF.NomListe.options[taille]=NomVariable;
```

Pour supprimer une option:

```
document.NomF.NomListe.options[i]=null;
```

XI. Les objets des propriétés et des fonctions prédéfinies:

1. L'objet window :

a. Méthodes de l'objet window

JavaScript dispose :
- Trois boites de dialogue : alert(), prompt() et confirm()
- Une minuterie : setTimeout() et clearTimeout().

b. Utilisation de la barre d'état :

Avec JavaScript, la barre d'état peut être utilisée pour afficher des messages.
Les propriétés à utiliser sont :
Status : valeur du texte affiché dans la barre d'état de la fenêtre.
DefaultStatus : valeur par défaut

c. Ouverture et fermeture de fenêtres

Les méthodes mises en œuvre sont :
open() : ouvre une nouvelle fenêtre.
close() : ferme la fenêtre en cours.

Syntaxe de window :

```
[window.]open("URL","nom_de_la_fenêtre","caractéristiques_de_la fenêtre")
```

Caractéristique	Description
toolbar=yes ou no	Affichage de la barre d'outils
status=yes ou no	Affichage de la barre d'état
menubar=yes ou no	Affichage de la barre de menus
scrollbars=yes ou no	Affichage des barres de défilement.
resizable=yes ou no	Dimensions de la fenêtre modifiables
width=x en pixels	Largeur de la fenêtre en pixels
height=y en pixels	Hauteur de la fenêtre en pixels

2. L'objet String :

a. La propriété length

Cette propriété indique la longueur de la chaîne de caractères.

Syntaxe : x = variable.length;

Exemples :

```
Ch="script";
n = Ch.length        -->   n contient 6
l = ("Javascript").length;    -->   l contient 10
```

b. La méthode CharAt()

Cette méthode permet d'accéder à un caractère isolé d'une chaîne.

La position du 1er caractère est 0 et celle du dernier est égale à la longueur (length) de la chaîne - 1.

```
chaine :      Javascript   (longueur = 10)
              ||||||||||
position :    0123456789   (longueur - 1)
```

Si la position indiquée est < 0 ou plus grande que la longueur – 1, JavaScript retourne une chaîne vide.

Syntaxe de charAt() :

variable = chaine.charAt(x); où x est un entier compris entre 0 et la longueur de la chaîne - 1.

Exemples :

```
Ch="Javascript";
C = Ch.charAt(4);      -->   C contient "s"
C = Ch.charAt(10);     -->   C contient "" (le vide)
```

c. La méthode indexOf()

Syntaxe : P = Ch.indexOf(Sch,y);

Cette méthode renvoie la position d'une sous chaîne (Sch) dans une chaîne (Ch) en commençant la recherche à la position y.

Si y n'est pas spécifié, la recherche commencera par défaut à la position 0.

Si la sous chaine n'est pas trouvé dans la chaîne la valeur retournée sera -1.

Exemples :

```
Ch = "Javascript"
Sch = "script"
x = Ch.indexOf(Sch);    -->   x  vaut 4 "
x = Ch.indexOf("@");    -->   x  vaut -1
```

d. La méthode substr()

Syntaxe : Sch = Ch.substr(P,N);

Cette méthode permet d'extraire d'une chaine donnée Ch, une sous chaine Sch à partir de la position P et d'une longueur N.

Exemples :

```
Ch = "Javascript"
x = Ch.substr(4,6);    -->   x contient la sous chaine "script"
x = Ch.substr(2,3);    -->   x contient la sous chaine "vas" (les positions 2,3 et 4)
```

3. L'objet Math :

Pour l'objet Math, on distingue les méthodes suivantes :

abs() :

```
x = -5
y = Math.abs(x);    -->   y vaut -5
```

sqrt() :

```
x = 9
y = Math.sqrt(x);    -->   y  vaut 3
```

round() : *arrondit le nombre à l'entier le plus proche*

```
x = 4,83
y = Math.round(x);   -->   y  vaut 5
```

random() : *retourne un nombre aléatoire entre 0 et 1*

```
x = Math.random();   -->   x  vaut  0,47513
```

eval() :

Cette fonction évalue une chaîne de caractère sous forme de valeur numérique. On peut stocker dans la chaîne des opérations numériques, des opérations de comparaison, des instructions et même des fonctions.

```
x = "4 + 8 * 2";
y = eval(x);    -->   y  vaut  20
z = eval (5>15);  -->  z  vaut  False
```

4. L'objet Date :

Pour l'objet Date, on distingue les méthodes suivantes :

new Date() :

Cette méthode sans arguments renvoie la date et l'heure courante.

Syntaxe : *d = new Date();*

Ces informations sont enregistrées par JavaScript sous le format : "Wed Sep 09 2009 14:49:06 GMT+0200"

getYear() :

Cette méthode retourne les deux derniers chiffres de l'année.

```
d = new Date();
an = d.getYear();    -->   dans l'exemple précedent an  vaut  109
```

getMonth() :

Retourne le mois sous forme d'un entier compris entre 0 et 11 (0 pour Janvier, …, 11 pour Décembre).

```
d = new Date();
m = d.getMonth();
```

getDate() :

Retourne le jour du mois sous forme d'un entier compris entre 1 et 31.

```
d = new Date();
 j = d.getDate();
```

getDay() :

Retourne le jour de la semaine sous forme d'un entier compris entre 0 et 6 (0 pour dimanche, 1 pour lundi, 2 pour mardi, …).

```
d = new Date();
 jour = d.getDay();
```

getHours() :

Retourne l'heure sous forme d'un entier compris entre 0 et 23.

```
d = new Date();
h = d.getHours();
```

getMinutes() :

Retourne les minutes sous forme d'un entier compris entre 0 et 59.

```
d = new Date();
m = d.getMinutes();
```

getSeconds() :

Retourne les secondes sous forme d'un entier compris entre 0 et 59.

```
d = new Date();
 s = d.getSeconds();
```