Université Mohammed Premier
Faculté des Sciences
Département d'Informatiques
Oujda

Master Mathématiques AA et AF

Technologies du Web : JavaScript

Mme Ibtissam.ARRASSEN
2018-2019

Introduction

JavaScript est l'un des 3 langages que tous développeur Web doit apprendre:

1. **HTML** pour définir le contenu des pages Web

2. **CSS** pour spécifier la disposition (layout) des pages Web

3. **JavaScript** pour programmer le comportement des pages Web

Ce cours décrit la façon dont JavaScript fonctionne avec HTML et CSS.

Introduction

- JavaScript est un langage de programmation qui peut être inclus dans des pages HTML.

  - Langage **interprété**: le navigateur exécute chacune des lignes de programme au fur et à mesure

#

  - Langages **compilés** et **traduits** en code machine pour être exécutés.

- JavaScript ne doit pas être confondu avec Java.

introduction

- Le Javascript est à ce jour utilisé majoritairement sur Internet.

- Permet de **dynamiser** une page HTML, en ajoutant des interactions avec l'utilisateur, des animations, de l'aide à la navigation, comme par exemple :

  - afficher/masquer du texte
  - faire défiler des images
  - créer un diaporama avec un aperçu "en grand" des images
  - créer des infobulles
  - ...

Les éléments du langage

- Variables et assignement
- Opérations
- La séquence
- Les boucles
- Fonctions et procédures

Variables et Assignement

- Javascript n'autorise la manipulation que de 4 types de données :

1. **Nombres** : entiers ou à virgules
2. **Chaînes de caractères (*string)*** *: une suite de caractères*
3. **Booléens** : des variables à deux états permettant de vérifier une condition :
   1. *false: lors d'un résultat faux*
   2. *true: si le résultat est vrai*
4. **Variables de type *null*** *: un mot caractéristique pour indiquer qu'il n'y a pas de données.*

- En Javascript, il n'y a pas besoin de déclarer le type de variables que l'on utilise.

Variables et Assignement

- Exemples :
  - *a = 3*
  - *b = “hello world”*
  - *c = ‘hello from Mars’*
  - *a = b*
  - *a = 3*4*3+2*

- Pour assigner une chaîne de caractères, on la place entre guillemets doubles ("…") ou simples ('…').

Variables et Assignement

- On attribue une valeur à une variable. La structure générale de l'assignement est :

  target = source.

- Il existe des notations abrégées pour certains types d'assignements. Par exemple :

Notation abrégée	Signification
x += y	x = x + y
x -= y	x = x – y
x++	x = x + 1
x--	x = x - 1

Opérations

- **Les opérateurs arithmétiques**

Les 4 opérations de base sont disponibles en mode texte :

- *+*
- *-*
- ***
- */*

- **Exemples :**
  - *somme = somme +1*
  - *b = c - d*
  - *somme = somme - 5*
  - *resultat = (3 + 5) * (23 / 4) – 3*

Opérations

- **Les opérateurs de comparaison**

Exprimer des conditions

égal à	==
différent de	*!=*
supérieur à	>
supérieur ou égal à	>=
inférieur à	<
inférieur ou égal à	<=

Opérations

- **Les opérateurs logiques**

Opérateur	Signification
ET logique	&&
OU logique	//
NON logique	!

La séquence

- Pour exécuter des instructions en séquence, il suffit d'écrire chaque instruction suivie d'un point-virgule :
  - *<instruction> ;*
  - *<instruction> ;*
  - *…*
  - *<instruction> ;*
- Exemple :
  - *a = 15;*
  - *b = 23;*
  - *c = 2 * a + b;*

Les choix (sélecteurs)

Choix avec une alternative

Formulation générale :

```
if (<condition>) {

<action>;
<action>;
…
}
```

Exemple :

```
if ( j == 5 ) {

Somme =
Somme +1;

}
```

Les choix (sélecteurs)

Choix avec deux alternatives

Formulation générale :

```
if (<condition>) {
<action>;
<action>;
…
} else {
<action>
<action>
…
}
```

Exemple :

```
if ( j == 5 ) {
Somme = Somme + 1 ;
}
else {
Somme = Somme - 1 ;
}
```

Les choix (sélecteurs)

Choix avec alternatives reliées

Formulation générale

*if (<**cond1**> **<operateur>** <**cond2**>) {*

<action>

<action>

…

}

Exemples

- *if (**NP< 4000 && NP >= 3900**) {*
  Canton = "Valais";
  }

- *if (!(**x<5 && y>7**)) {*
  resultat = "correct";
  }

Les choix (sélecteurs)

Formulation :

```
if (<condition1>) {
    <action>
    }
else {
    if (<condition2>) {
    <action>
    <action>
    …
    }
    else
    {
    <action>
    <action>
    …
    }
}
```

Choix avec conditions imbriquées

Exemple :

```
if ( reponse ==1 ) {
cadeau = "cigarettes";
}
else {
        if ( reponse==3 ) {
                cadeau = "fleurs" ;
        }
        else {
                cadeau = "chaussettes" ;
        }
}
```

Les boucles : While

Formulation générale :

```
while ( <condition> ) {
    <action>;
    ...
}
```

Exécute les <actions> aussi longtemps que la <condition> est vraie.

Si la condition est fausse au début, aucune instruction n'est exécutée.

Exemple :

```
chiffre = 0 ;
somme = 0 ;
while ( chiffre < 5 ) {
    chiffre = chiffre + 1;
    somme = somme + chiffre;
}
```

Les boucles : do ... while

Formulation générale :

```
do {
    <action>
    ...
} while ( <condition> )
```

Exécute les <actions> aussi longtemps que la <condition> est vraie.

Exemple :

```
chiffre = 0 ;
somme = 0 ;
do {
    chiffre = chiffre + 1 ;
    somme = somme + chiffre;
} while ( chiffre< 5 )
```

Si la condition est fausse au début, les instructions seront quand même exécutées une seule fois.

Les boucles : for

Formulation générale :

```
for (   <expression de depart> ;
                <condition de continuation> ;
                                <incrementation>    )  {
<action>
...
}
```

Exemple:

```
somme = 0 ;
for ( chiffre = 1 ; chiffre <= 5 ; chiffre = chiffre + 1 ) {
                somme = somme + chiffre ;
}
```

Les boucles : for

Signification : :

- Tant que la condition de continuation est vraie :
  - en partant de l'expression de départ (*chiffre = 1) on exécute le contenu des accolades;*
  - la variable (chiffre) est incrémentée (*chiffre = chiffre + 1) et on exécute le contenu des* accolades autant de fois que nécessaire;
  - la boucle ne prend fin que lorsque la condition de continuation devient fausse.

Fonctions et procédures

- Fonction: un sous-programme qui permet d'effectuer un ensemble d'instruction par simple appel de la fonction dans le corps du programme principal.

- Les fonctions et les procédures permettent d'exécuter une série d'instructions dans plusieurs parties du programme, cela permet une simplicité du code et donc une taille de programme minimale.

- Dans JavaScript, les fonctions et les procédures sont définies par le mot clé ***function***

Fonctions et procédures

- la fonction retourne une valeur (numérique, booléen etc.),
  # à une procédure.

- Ce retour de valeur se fait par le mot clé ***return.***

- Une fonction doit être tout d'abord déclarée.

- Une fonction doit avoir un nom, des arguments et les instructions qu'elle contient.

- La déclaration d'une fonction se fait grâce au mot clé *function selon la syntaxe* suivante:

```
function  nom(parametre1, parametre2,  …,  parametreN) {
....
}
```

Fonctions et procédures

Exemple :

```
function  carre(nombre) {
    resultat = nombre * nombre ;
    return  (resultat)
}
```

Méthodes

- Une méthode est une fonction associée à un objet : une action que l'on peut faire exécuter à un objet.

- Les méthodes des objets du navigateur sont des fonctions définies à l'avance par les normes HTML, on ne peut donc pas les modifier, il est toutefois possible de créer une méthode personnelle pour un objet que l'on a créé soi-même.

- une page HTML est composée d'un objet appelé ***document*.**

- *Il a la méthode* ***write()*** *qui lui est associée et qui permet d'*afficher du texte dans la page HTML.

- Dans le cas de la méthode write(), l'appel se fait comme suit :

window.document.write()

Javascript et HTML

1. La balise ***<SCRIPT>*.**
2. Affichage et Introduction des données : ***alert*** *et* ***prompt*.**
3. Les objets.
4. Les événements.

La balise *<SCRIPT>*

- Le code javascript doit être placé à l'intérieur de la balise <script> … </script>

- Une page HTML peut contenir plusieurs balises *<script>, mais elles ne doivent pas être* imbriquées.

- On peut placer une balise ***<script> soit dans l'entête <head>, soit dans le corps <body> de*** la page HTML.

- On placera de préférence une balise ***<script> contenant les procédures et*** les fonctions dans l'entête, car cela permet qu'elles soient chargées avant le reste de la page.

La balise *<SCRIPT>*

- Comme il y'a d'autres langages comme JavaScript, il est nécessaire d'ajouter l'attribut *LANGUAGE="JavaScript" dans la balise* ***<SCRIPT>.***

```
<SCRIPT LANGUAGE="JavaScript">
    …code JavaScript…
</SCRIPT>
```

- Il est aussi possible d'ajouter des scripts à une page HTML à partir d'un fichier. Dans ce cas, on ajoute à la balise *<script> un paramètre précisant le nom du fichier contenant les* scripts :

<SCRIPT LANGUAGE="JavaScript" ***SRC="Nom_fichier.js"****>*

Affichage et introduction des données : *alert et prompt*

- La fonction ***alert*** *sert à afficher une valeur.*

Exemple:

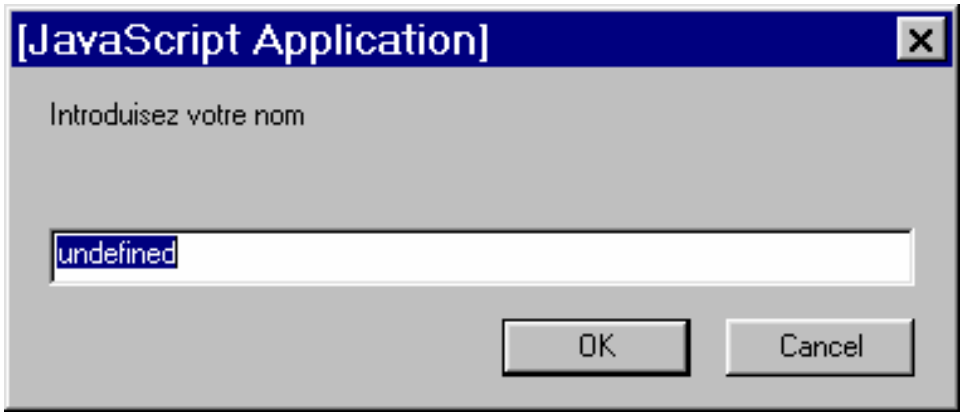
alert ("Hello World !");

Affichage et introduction des données : *alert et prompt*

- La fonction *prompt sert à lire une valeur.*

Exemple:

- ***x = prompt("Introduisez votre nom");***

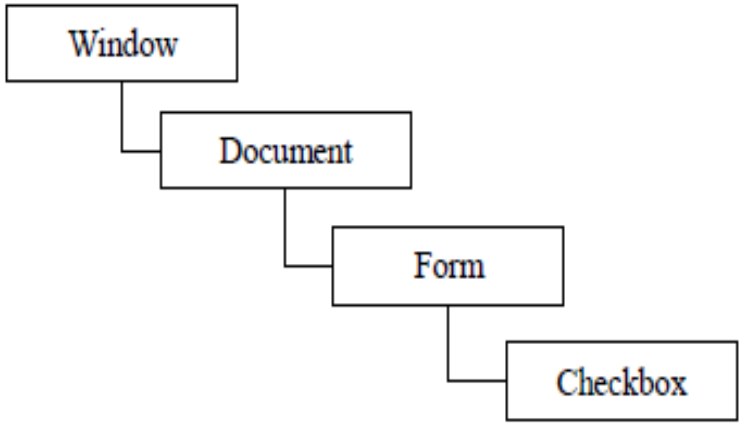


- Après avoir cliqué sur **OK, la variable *x contient la chaîne de caractères qui a été introduite.***

Les objets

- Javascript traite les éléments qui s'affichent dans votre navigateur comme des objets, càd des éléments classés selon :
  - une hiérarchie pour pouvoir les désigner précisément
  - auxquels on associe des propriétés

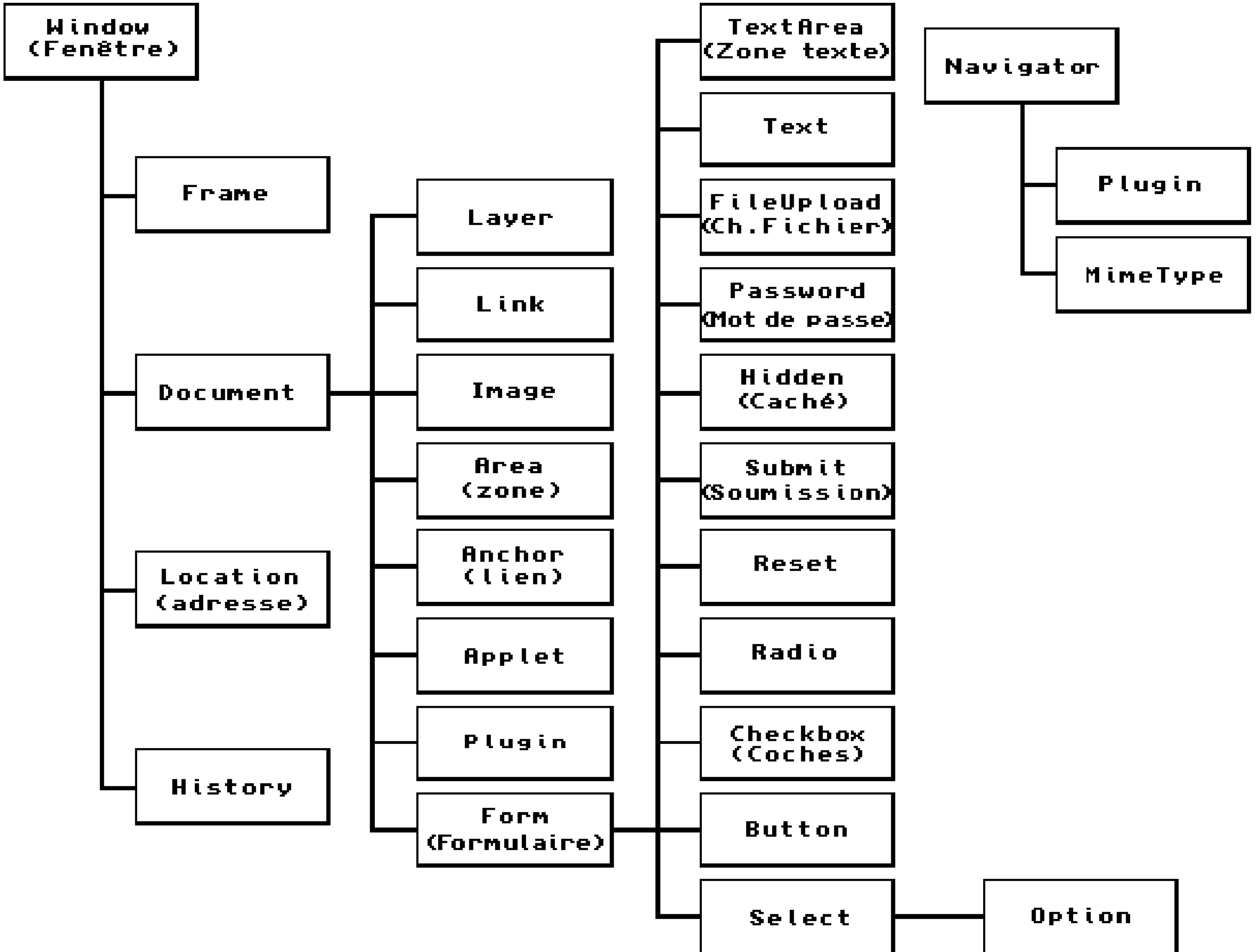
- Par exemple, pour atteindre un bouton à l'intérieur d'un formulaire, la hiérarchie est :



Objet

- Les objets contiennent 3 choses distinctes :
  - Un constructeur
  - Des propriétés
  - Des méthodes

La hiérarchie des objets est la suivante :



Objets

- L'accès aux objets se fait par une notation par points.

- D'autre part, comme il peut y avoir plusieurs formulaires et plusieurs boutons dans chaque formulaire,

➔certains objets sont automatiquement numérotés.

Exemple: accéder au premier bouton du premier formulaire d'une page web se fait par la notation suivante :

window . document . forms[0] . checkbox[0]

Les événements

- Les événements sont des actions de l'utilisateur qui vont pouvoir donner lieu à une interactivité.

- **Exemple** : L'événement par excellence est le clic de souris, car c'est le seul que le HTML gère.

- Grâce au Javascript il est possible d'associer des fonctions, des méthodes à des événements :

    - passage de la souris au-dessus d'une zone,
    - changement d'une valeur dans un champ, etc.

Evénement

- les gestionnaires d'événements permettent **d'associer une action à un événement**.

- Syntaxe :
  - ***onEvenement****="Action_Javascript_ou_Fonction()";*

- Les gestionnaires d'événements sont associés à des objets, et leur code s'insère dans la balise de ceux-ci.

- Chaque événement ne peut pas être associé à n'importe quel objet.

- Ex: l'événement ***OnChange*** *ne peut pas s'appliquer à un lien hypertexte.*

Objets et événements associés

Le tableau ci-dessous résume les objets et les événements associés :

Objet	Evénements associables
Lien hypertexte	*onClick, onMouseOver, onMouseOut*
Page du navigateur	*onLoad, onUnload*
Bouton, Case à cocher, Bouton radio, Bouton Reset	*onClick*
Liste de sélection d'un formulaire	*onBlur, onChange, onFocus*
Bouton Submit	*onSubmit*
Champ de texte et zone de texte	*onBlur, onChange, onFocus, onSelect*

Evénement	*Objets affectés*	*Description*
onClick	Boutons, boutons radio, boutons submit et reset, liens	S'exécute quand on clique dans ou sur un élément
onBlur	Fenêtres et tous les éléments de formulaire	S'exécute quand on quitte la fenêtre ou un objet de formulaire
onChange	Champs texte, zones texte, listes de sélection	S'exécute quand un élément de formulaire est modifié.
onError	Images, fenêtres	S'exécute quand le chargement de l'image ou de la fenêtre provoque une erreur