Javascript
1° PARTIE : LES BASES

1. Définition
2. Motivations
3. La balise <SCRIPT>
4. Insertion de code javascript
5. Emplacement du code javascript
6. Exemple
7. Les variables
8. Expressions et opérateurs
9. Structures de contrôles
10. Les fonctions
11. Fonctions prédéfinies

1 . Définition

Javascript est un langage de script, qui incorporé aux balises HTML, permet d'améliorer la présentation et l'interactivité des pages Web côté client.

Javascript a été initialement développé par Netscape (*LiveScript*), puis adopté en 1995, par la société SUN (qui développa aussi Java), pour prendre le nom de Javascript.

2 . Motivations

- Ajout du contrôle au niveau du client

• Exemple : contrôle des entrées d'un formulaire avant son envoi.

- Scripts embarqués dans des documents HTML

• Contrôle des ressources du client (documents, formulaires, …) DONC moins d'intervention du Serveur WWW
• Génération dynamique de documents DHTML (*layers*)
• Éviter les GIFs animés et les animations Shockwave

3 . La balise <SCRIPT>

- La balise <SCRIPT> permet d'indiquer au navigateur le début d'un script. La fin du script sera indiqué par la balise </SCRIPT>.

- Exemple d'insertion d'un code js dans une page HTML :

```
<SCRIPT LANGAGE="JavaScript">
… Instruction; ...
</SCRIPT>
```

4 . Insertion du code js

- **Interne** au document (on peut insérer du code javascript à plusieurs endroits dans un document HTML).

- **Externe** au document : le code javascript se trouve dans un fichier séparé portant le plus souvent l'extension **`.js`**. On devra préciser dans la balise <SCRIPT> le chemin du fichier :

```
<SCRIPT LANGAGE="JavaScript" SCR="script.js">
//inclut le fichier script.js puis l'exécute
</SCRIPT>
```

- Cas particuliers …

4 . Insertion du code js
(CAS PARTICULIERS)

Dans certains cas, l'usage de la balise <SCRIPT> n'est pas obligatoire :

• Cas des événements où il faut simplement insérer le code à l'intérieur de la balise HTML comme un attribut de celle-ci :

```
<BODY onLoad="message()">
```

• Cas des liens HREF : on peut insérer du code javascript dans un lien **`href`** en précisant le mot clé **`javascript:`** :

```
<A href="javascript:message()">une fonction js</A>
```

5 . Emplacement du code js

Il suffit de respecter les deux principes spécifiques aux scripts :

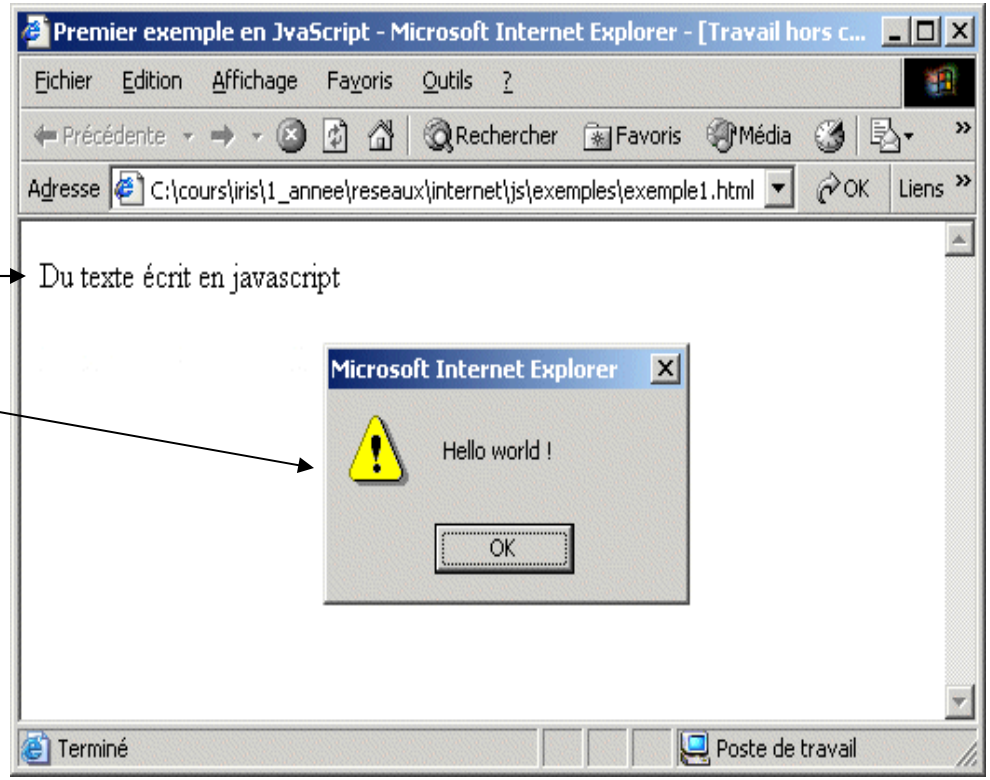
- N'importe où
- Mais là où il faut

Les instructions ne pourront s'exécuter que, si le navigateur possède à ce moment précis, tous les éléments nécessaires à son exécution.

✓ Le code javascript est généralement inséré entre les balises <HEAD> et </HEAD>.

6 . Exemple

```
<HTML>

<HEAD>
<TITLE>Premier exemple en JavaScript</TITLE>

<SCRIPT LANGAGE="Javascript">
document.write("<P>Du texte écrit en
javascript</P>");
alert("Hello world !");
</SCRIPT>

</HEAD>

<BODY>
<P>Du texte écrit en HTML</P>
</BODY>

</HTML>
```



7 . Les variables

- Pas de déclaration préalable des variables

```
nbr = 10; //un entier
fl = 3.141; //un réel
str1 = "L'étoile"; //une chaîne de caractère
str2 = 'brille'; //une chaîne de caractère
```

  Il existe aussi le type booléen (*true* ou *false*)

- Portée des variables :
  - ✓ Local (uniquement dans le script ou la fonction) : `var vloc = 0;`
  - ✓ Global (en tout point du document) : `vglob = 0;`

8 . Expressions et Opérateurs

- Expressions
  - **Arithmétique :** `(3+4) * (56.7 / 89)`
  - **Chaîne :** `"Une étoile" + " " + "filante"` (**concaténation**)
  - **Logique :** `h2o = (temp<100) ? "eau" : "vapeur";`

- Opérateurs
  - **Affectations :** `+=, -=, *=, /=, %=, &=, |=, <<=, >>=`
  - **Comparaisons :** `==, !=, <, <=, >, >=`
  - **Arithmétiques :** `%, ++, --`
  - **Logiques :** `&&, ||, !`
  - **Bit :** `&, |, ^ (XOR), ~ (NON), <<, >>`

9 . Structures de contrôle

La syntaxe est identique à celle du C/C++ :

- Bloc : `{ }`

- **`if else, switch case, for, while, break, continue, do while, for in`**

10 . Les fonctions

- Définition et Appel :

```
function nomfonction( param1, ..., paramN)
{
	// code JavaScript …
	return variable_ou_valeur ;
}
var res = nomfonction(var1, val2, varN) ;
```

→ **<u>Remarque :</u> passage des paramètres par valeur**

- Exemple :

```
function isHumanAge(age)
{
	if ((age < 0) || (age > 120)) { return false  ; }
	else { return true  ; }
}

if(!isHumanAge(age))
{
	alert("Vous ne pouvez pas avoir " + age + " ans !");
}
```

10 . Les fonctions
(arguments variables)

```
function somme()
{
   var argv = somme.arguments;
   var argc = somme.arguments.length;
   var result=0;
   for(var i=0 ; i < argc ; i++)
   { result += argv[i]; }
   return result;
}
```

somme(1,2,3) retournera 6

somme(2) retournera 2

11 . Fonctions prédéfinies

- **`parseInt, parseFloat`** : retourne la représentation entière ou réelle de l'argument

```
parseInt("12", 10) ), parseInt("1100", 2) ), parseInt("0xC", 16) ) // retournent 12
parseFloat("1.23") // 1.23
parseFloat("1,23") // 1
parseFloat("1abc23") // 1
floatValue= parseFloat("abc789"); // NaN
if (isNaN(floatValue)) { alert("notFloat ! "); }
else { alert(floatValue); }
```

- **`typeof`** : c'est un opérateur qui permet de lire le type de la variable

```
type = typeof variable;
```

Javascript connaît 6 types différents : undefined, boolean, number, string, function et object.

- **`escape(), unescape()`** : utilisées pour le codage des **URL** ou des **Cookies**

```
escape("#") // retourne %23
unescape("%23") // retourne #
```