Les bases de la programmation en JavaScript

Dr N. BAME

Javascript

- Langage de script à base d'objet **multiplateforme** standardisé par le ECMAScript
- Créé par Netscape (1995, Brendan Eich) pour le traitement local des événements provenant du client (*LiveScript)*

- Insertion d'instructions de programmation dans le code des pages HTML

- **Exécution** du code **sur le client**
  - ✓ Améliorer **l'interactivité**
  - ✓ Économie du **coût** de communication
  - ✓ Pages **dynamiques** (animation, personnalisation)

- Programmation d'action en fonction **d'événement** de l'utilisateur
  - ✓ Contrôle de saisie
  - ✓ Calcul sur la machine du client

Javascript

- **Noyau javaScript**
  - ✓Objets **prédéfinis**, **opérateurs**, **fonctions**, **structures**, …
  - ✓Ensemble **d'objets associés au navigateur**
    - Fenêtres,
    - Documents,
    - Images, …

Javascript - Java

JavaScript n'est pas java

javaScript

- interprété
- A base d'objets
- Code intégré dans HTML (visible)
- Typage faible
- N'existe pas en dehors du web

Java

- compilé
- Orienté objet
- Code dans applets (non visible)
- Typage fort
- Langage à part entière

Ecriture du code javascript

Deux méthodes

1. Utilisation de la balise **<script> … </script>**
   - ✓ Déclaration de fonction dans l'entête (entre <head> et </head>
   - ✓ Appel de fonction ou exécution de code dans <body>…</body>
   - ✓ Insertion d'un **fichier javascript externe**

2. Utilisation de **nouveaux attributs** de balise pour la gestion d'événement d'utilisateur
   - ✓ <balise **onEvenement**="code javaScript">

Ecriture du code javascript

```
<html>
    <head>
        ...
        <script language= "javascript ">
                function saluer(){window.alert('Bonjour');}
        </script>
    </head>
    <body>
        <script language= "javascript" >
                document.write("Pour être saluer");
        </script>
        <br/><input type= "button" onClick= "javascript:saluer();"
                        value= "cliquez ici" >
        </body>
</html>
```

Ecriture du code javascript

```
<script langage="languageName" src="fileName"></script>
```

- ✓ langageName : language de script utilisé, **par défaut javascript**
- ✓ fileName : permet d'insérer du code à partir d'un fichier.

- **Exemple**

```
<script langage="javascript" src="authent.js"></script>
```

Le langage javaScript

- JavaScript est un langage **faiblement typés**
  - on n'indique pas le type des variables lors de la déclaration.
    - Lors le la déclaration des variables, le type est fixé implicitement par le type de la donnée affectée à la variable.
- Opérateurs et instructions comme en C
- **Méthodes**
  - Globales (associées à tous les objets)
  - Fonctions prédéfinies par l'utilisateur
- **Objets**
  - **prédéfinis** (String, Date, …)
  - Liés à l'environnement (window, document, …)
- Commentaire comme en C
- Séparateur d'instructions : ; (retour chariot)

Opérateurs

- Mêmes opérateur qu'en C (ou PHP).
- Remarques : l'opérateur **+** permet de concaténer des chaînes de caractères

Variables

- Sensible à la **casse**
- **Déclaration explicite** de variable
  - Optionnelle mais fortement conseillée
  - Avec l'instruction **var**
  - Pas de précision de type à la déclaration
  - Initialisation possible lors de la déclaration, sinon la valeur undefined

- Notion de variables locales et globales :
  - locales à une fonction
  - globales au document HTML

Variables et types

- Le typage a lieu lors de l'initialisation ou d'une affectation : langage dynamiquement typé

- Le type d'une variable peut changer si on lui affecte une valeur d'un autre type

- Types de données simples
  - Number : nombre
    - Entier, réel
  - String : chaîne de caractères
    - 'ma chaine' ou "ma chaine"
  - Boolean : booléen
    - True, False

- Constantes : même syntaxe qu'en C

Conversion de type

- Type String = type dominant
- JavaScript fait des conversions implicites selon les besoins
- Exemples
  - N=12;//N est numérique
  - T="34" // T est une chaine
  - X=N+T // X est la chaine "1234"

- Il existe des types particuliers :
  - null, undefined, object, function
- Et des nombres particuliers :
  - Infinity, -Infinity, NaN (Not a Number)
- La fonction typeof() retourne le type de l'entité
- Conversion possibles avec : **parseInt(), parseFloat(), parseString(),** …

Fonctions

```
function nomFonction (arg1,…,argN)
{
    Instruction1;
    …
    InstructionM;
    [return valeur;]
}
```

- ✓ Arguments **non typés**
- ✓ Nombre d'arguments non fixé par la déclaration
  - Tableaux : arguments

Appels

```
var message = prompt('Entrez un texte');
document.write(message );
var myMar=nomFonction(arg1,…,argN);
```

Fonctions : exemples

- alert()
- Prompt()
- Confirm()
- parseInt()
- write()

Fonctions **anonymes**

- Fonctions particulières qui sont extrêmement importantes en Javascript
  - Manipulation des objets, des évènements, des variables statiques

```
function (arguments) {
    // Le code de votre fonction anonyme
}
```

- Exécution:
  - assigner la fonction à une variable.

```
var saluer = function() {
alert('Bonjour !');
};
```

  - Utilisation de la variable

```
saluer();
```

Structures de contrôles et boucles

- Même fonctionnement qu'en C

  - ✓ **if**
  - ✓ **switch**
  - ✓ **while**
  - ✓ **for**
  - ✓ **do … while**

Tableaux

- Déclaration

```
var etudiants = ['Moussa', 'Mamadou', 'Abdoulaye'];
var classes = new Array('L3Pro', 'L3Class', 'M1SIR‘);
```

- Accès aux éléments d’un tableau

```
nomTableau[index];
```

- Ajout et suppression d’éléments
  - ✓ **push()** : **ajouter** d’un ensemble d’éléments **à la fin** tableau
    - ✓ classes.**push**(‘M1RETEL’,’L3BioINF’);
  - ✓ **unshift()** : **ajouter** d’un ensemble d’éléments **en début** du tableau
  - ✓ **pop()** : **supprimer** le **dernier** élément du tableau
  - ✓ **shift()** : **supprimer** le **premier** élément du tableau

Objet littéral (tableau associatif)

```
var tab={
    nom:'Ndiaye',
    prenom:'Moussa'
    };
document.write(tab['nom']+' '+tab['prenom']);
```

- Parcours avec **for in**

```
for (var key in tab)
{
    document.write('<br>'+tab[key]);
}
```

Les objets

- Création

```
var myObjet = {
        property1 : value1,
        property2 : value2,
        …
        propertyN : valueN
};
```

- Accès à la valeur d'une propriété

```
myObjet.propertyi
```

ou `myObjet['propertyi']`

- Ajout d'une propriété

```
myObjet.Newproperty=value;
```

- Suppression d'une propriété

```
delete myObjet.property;
```

Exemple

```
function creerImaginaire(re, im)
{
    return { reel:re, imaginaire: im };
}

z=creerImaginaire(5, 9);
document.write(z.reel+' '+z.imaginaire+'i');
```

Exercice

- Définir un script js qui demande votre nom, votre prénom, votre age (valide : entre 17 et 35) et votre genre pour vous afficher un message de bienvenue personnalisé
- Définition et utilisation d'une fonction
- Définition d'un objet etudiant avec des valeurs
  - Afficher/modifier
  - Ajout de champs
  - Utilisation for in