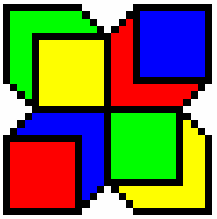
Le Langage



JavaScript

- Types de bases
  - ✓ **Boolean** (Booléen) : peut prendre les valeurs **true** (vrai) et **false** (faux)
  - ✓ **Number** (Nombre) : pour tous les nombres entiers et réels
  - ✓ **String** (Chaîne) : pour les chaînes de caractères, il n'existe pas de type pour représenter un seul caractère
- Déclaration et Initialisation

```
var iToto = 10;
var bVraiFaux = true;
var fe = 3.00e+8, fPi = 3.14, fPi2 = 3.14 * 3.14;
var a = 'a', nligne = '\n';
var phrase = "et voici javaScript";
```

- Variables et données
  - ✓ Une variable peut contenir des données de type quelconque
  - ✓ Le type d'une variable change en même temps que celui des variables qu'elles contiennent

=> Une variable n'a pas de type prédéfini (typage faible)

La Grammaire
Opérateurs

- assignement
  - ✓ `lvalue = rvalue; //@ mémoire`
- opérateurs arithmétiques
  - ✓ `+ ,- , /, *, %( modulo)`
- précédence
  - ✓ `a = x + y - 2/2 + z;`
  - ✓ Unaire, Arith., relationnelle, logique, cond., assignement
- opérateurs unaires
  - ✓ `x = -a; x = a * (-b);`
- auto incrémentation
  - ✓ `++i //(i = i + 1); i++;`
  - ✓ `--i //(i = i - 1); i--;`
  - ✓ `x += 4 //(x = x+4)`
- opérateurs relationnelles

  generent un résultat booleén
  - ✓ `<, >, >=, ==, !=`
- opérateurs logiques

  Sont utilisés avec les opérateurs relationnels
  - ✓ Et logique (&&)
  - ✓ Ou logique (||)
  - ✓ Non logique(!)

  `(a < b) || ( c == d)`
- opérateur sur les bits
  - ✓ AND(&) OR(|) XOR(^) NOT(~)
  - ✓ gauche (<<), droite (>>)
  - ✓ gauche zero extension (>>>)

  traite les nombres comme si il s'agissait de chaînes de 32 bits.

La grammaire
Tableaux simples

- Tableau : « **Array** »
  - ✓ Variable qui permet de contenir plusieurs données au même moment.
- Opérateur spéciale **"new"**
  - ✓ permet de construire un tableau

  var tab1 = new Array;

  var tab2 = new Array(5);

  var tab3= new Array('RC',2, 'EV',10);
- le champ **`length`**
  - ✓ est une variable qui donne la taille du tableau

  tab3.length est égale à 4 (de 0 à 3)

```
<HTML>
<BODY>
<SCRIPT LANGUAGE=JavaScript>

/* Exemple de Création de tableaux par
   l'operateur new en JavaScript
*/

// déclaration
var tab = new Array();

// initialisation
tab[0] = "Caroline";
tab[1] = "Jean-Christophe";
tab[2] = "Remy";

// affichage du contenu
Document.write ("tab[0]:"+tab[0]+<BR>);
Document.write ("tab[1]:"+tab[1]+<BR>);
Document.write ("tab[2]:"+tab[2]+<BR>);

// modification d'un élément du tableau
tab[2] = "RC";
</SCRIPT>
</BODY>
</HTML>
```

La grammaire
Tableaux multi-dimensionnels

- Gestion de tableaux de tableaux
- Un tableau est un objet dans le langage JavaScript
- Les tableaux ne sont pas forcément rectangulaires

```
// tableau multi-dimensionnels
...
var jeuEchec = new Array();
jeuEchec[0] = new Array();
jeuEchec[0][0] = "Tour";
jeuEchec[0][1] = "Cavalier";
...
jeuEchec[1] = new Array();
jeuEchec[1][0] = "Pion";
jeuEchec[1][1] = "Pion";
...
```

```
<HTML><HEAD><SCRIPT>
function tab() {        //tableau triangulaire
   var tri = new Array(8), i;
   for (i=0; i < tri.length; i++) {
     tri[i] = new Array (i+1);
     for (j=0; j < i+1; j++) {
       tri[i][j] = i+j;
       document.write("tri[" + i + "][" + j + "]=" + tri[i][j]+ " " );
     } document.write("<BR>"); }
}
</SCRIPT></HEAD><BODY ONLOAD="tab()"></BODY></HTML>
```

La Grammaire
Structures de contrôle

- *Déclarations conditionnelles*

```
if (condition){
  instructions;
} [else if (condition) {
       instructions;}]
  [else {
    instructions;}]
```

```
if (i<1)
        r = i * 100;
else    r = i * 10;

// i<10 ? r=i*100 : r=i*10;
```

```
switch (expression) {
case expression :
  instruction;
[case expression :
  instruction;
...
default :
  instructions;]
}
```

```
var langage = prompt(" … ");
switch (langage) {
    Case "Java":
    case "Langage C":
            compilation = true;
            break;

    case "JavaScript" :
            compilation = false;
}
```

La Grammaire
Structures de contrôle (2)

- **Boucles**
  - ✓ *Tantque*

```
while (condition) {
  ... //instructions
}
```

  - ✓ *Répéter*

```
do {
  ... //instructions
} while (condition);
```

  - ✓ *Pour*

```
for(init, condition,
  incrémentation) {
  ... //instructions
}

for(index in unTableau) {
  ... //instructions
}
```

```
// Afficher les multiple de 8 < à 100
var indexBoucle = 0;
While (++indexBoucle < 100) {
   // est-ce un multiple de 8 ?
   if (indexBoucle % 8 == 0)
     document.write (indexBoucle+" ");
}

var monTableau = new Array("A","B","C");
var index;
for (index = 0; index<3; index++) {
   document.write (monTableau[index]);
}

// equivalent à :
for (index in monTableau) {
   document.write (monTableau[index]);
}
```


La Grammaire
Les fonctions

- Objectif
  - ✓ Regrouper dans un bloc, un ensemble d'instruction réalisant une opération « complexe »
  - ✓ Ce bloc est identifier par un nom
  - ✓ Il peut être invoqué (exécuté) autant de fois que nécessaire au sein d'un code javaScript
  - ✓ Ceci évite de réécrire plusieurs fois un code semblable dans un programme
- Syntaxe

Function nomDeLaFunction (listeDArguments) **{**

/* Liste d'instructions

Éventuellement l'instruction **return** qui permet de sortir de la fonction avant la fin de la liste d'instructions

*/

}

La Grammaire
Les fonctions (2)


```
<html>
<head>
    <title> ..... </title>
    <script language="JavaScript">
        // déclaration des variables
        var var1;
        var var2;
        // début des fonctions
        function NomDeLaFonction (paramètre_1, … , paramètre_i) {
                instruction_1; …;  instruction_n;
         }
    </script>
</head>
<body>
…
</body>
</html>
```


La Grammaire
Les Objets JavaSript

- Objectif
  - ✓ Regroupement d'un ensemble de variables et de fonctions à l'intérieur d'un ensemble organisé et nommé qui pourra facilement être réutilisé
- Exemple

  Prenons les objets du monde réel : un ordinateur par exemple
  - ✓ Comment définissons nous un ordinateur ?
    - = ensemble de propriétés { marque, processeur, RAM, …}
  - ✓ Comment utilisaons nous notre ordinateur ?
    - = ensemble de fonctions {mettre sous tension, éteindre, cliquer, …}
- Utilité
  - ✓ Il est possible de créer autant **d'instances** d'objet que l'on veut avec les mêmes caratéristiques (pripriétés et fonctions)
- Différence avec les véritables LOO C++, Java, etc.
  - ✓ La définition d'objets dans de tels langages obéit à des règles très précises qui ne s'applique pas à JavaScript qui n'impose pas une grande rigueur d'écriture.
  - ✓ Pas de Polymorphisme, de sécurité d'accès aux fonctions, …

Les Objets (2)

Création d'un objet, prototype et constructeur

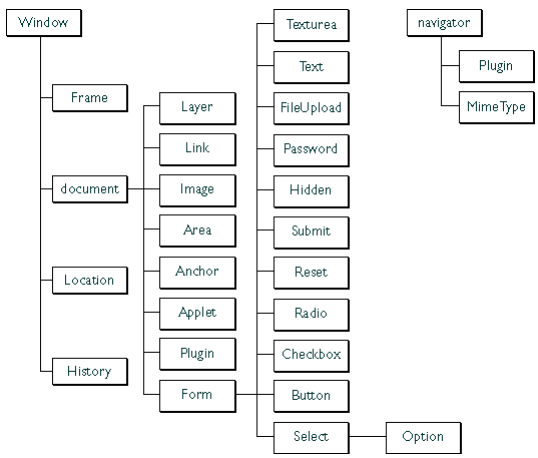
```
<script Language=JavaScript>
    function Stagiaire(nom, experience){ // constructeur
          this.nom = nom;
          this.experience = experience; // de 1(débutant) à 5(expert)
          this.dernierStage = "JavaScript";
    }
    function display() {
          document.write(this.nom); // variable spéciale this
    }
    Stagiaire.prototype.ville = "quatre borne";
    Stagiaire.prototype.universite = "Maurice";
    Stagiaire.prototype.afficher =  display;

    var stagiaire1 = new Stagiaire("Islam", 1);
    var stagiaire2 = new Stagiaire("Jean", 5);

    stagiaire1.afficher();
    document.write(stagiaire2.nom + " " stagiaire2.ville);
</script>
```

Construteur d'objets de type Stagiaire toujours utilisé avec le mot clé new. Il attribue automatiquement des propriétés à l'objet

La propriété prototype dote automatiquement tout nouvel objet d'un certain nombre de propriétés

Créer un objet Stagiaire mémorisé dans la variable stagiaire1

Les Objets « navigateur » utilisés par JavaScript



Exemple d'utilisation d'objets du navigateur


```
<html><head><script>
function tab() {
 with (document) {
  var tri = new Array(8);   var i;
  for (i=0; i < tri.length; i++) {
   write("<A HREF='http://page" + i+ ".html'> tri[" + i + "]</A> ");
   tri[i] = new Array (i+1);
   for (j=0; j < i+1; j++) {
        tri[i][j] = i+j;
        write("tri[" + i + "][" + j + "]=" + tri[i][j]+ " " );
   }
   write("<BR>");
  }
  write("<BR><HR><BR>");

  for (i in document.links)
   if (!isNaN(i))
        write("lien No " + i + " : " + document.links[i]+ "<BR>");
 }
}
</script></head><body ONLOAD="tab()"></BODY></body></html>
```