IFT-6800, Automne 2016

Cours 9: Introduction à Javascript

Louis Salvail
André-Aisenstadt, #3369
salvail@iro.umontreal.ca

Langages de programmation évolués

- Les langages de programmation évolués permettent de programmer votre ordinateur en faisant abstraction de son langage machine.
- Il devient possible d'écrire des programmes qui peuvent être exécutés sur la plupart des machines indépendamment de son architecture propre.
- La plupart des langages évolués ont des propriétés propres qui permettent de faciliter la programmation par rapport à d'autres langages dans certains contextes:
  - **SQL** facilite la mise au point d'applications qui doivent consulter et mettre en place de grandes quantités de données. Il ne s'agit pas d'un langage de programmation au sens traditionnel. Toutes ces applications sont du même type.
  - **C/C++** facilite le développement de SE, de compilateurs, etc… Étant près du langage machine, il permet de mettre au point des applications de bas niveau (comme les SE, les compilateurs, les pilotes,…) qui sont efficaces. **C++** est une version de **C** du type orienté objet.
  - **JavaScript** permet le développement d'applications attachées à des pages web pour les rendre dynamiques. Mis au point pour être facile à utiliser par des non experts. Il fait partie de l'ensemble des langages que l'on désigne comme des scripts. Nous y reviendrons.
  - **Java** est un langage qui permet le prototypage rapide. Idéal dans les situations où des applications doivent être produites rapidement. Très portable, facile à apprivoiser. Il est orienté objet.
  - etc……

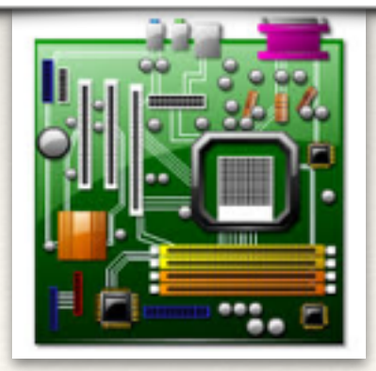
Langages évolués sur votre machine

C est un langage compilé

Un programme écrit en **C**

Compilateur qui traduit le programme C en langage machine

Programme en langage machine

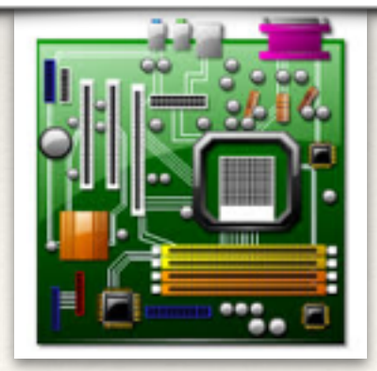


JavaScript est un langage interprété

Un programme écrit en **JavaScript**

Un interpréteur au moment de l'exécution transforme chaque commande en instruction machine

Exécute instructions machines



Langages de script

- Un langage de script permet d'écrire des programmes pour un environnement d'exécution spécialisé qui interprète et automatise l'exécution de tâches qui pourraient être exécutées une à la suite de l'autre dans l'environnement.
- Les environnements qui peuvent être automatisés par des scripts sont
  - des applications logiciels,
  - des pages web,
  - des navigateurs web,
  - les shell d'un SE, …
- Un langage de script est un langage appliqué à un domaine spécifique dans un environnement particulier. On parle souvent de langage de très haut niveau: ils opèrent d'une façon abstraite.

Javascript

- Un langage évolué mis au point pour programmer des pages web interactives du côté client et, plus récemment, pour programmer des serveurs web.
- Javascript est un langage de script, il est interprété: sa traduction en langage machine se fait au moment de l'exécution.
- C'est un langage du type orienté objet qui vise le prototypage rapide et dont la syntaxe est facile à apprendre.
- Langage créé en 1995 par Bernard Eich pour le compte de Netscape, inspiré des langages Java et Python en les simplifiant, utilisable par des débutants.

Javascript (II)

- L'environnement privilégié dans lequel Javascript opère est une page web.
  - `<script> ici le script javascript </script>`
  - Un script est donc indiqué par une paire de balises HTML que vous placez habituellement entre les balises `<head>` et `</head>`.
- L'interpréteur javascript est le navigateur lui-même.
- Puisque l'environnement d'un programme javascript est la page dans laquelle il est inclus, il ne voit rien d'autre. Il n'a accès aux périphériques de l'ordinateur qu'à travers la page web. Il ne peut écrire sur le disque directement.

Les variables en javascript

- La première chose à éclaircir est la déclaration des variables en javascript. Une variable est un espace mémoire réservé pour le rangement d'information pertinente dans le programme.
- Les variables sont déclarées avec le mot clé `var`:
  - `var a;`
  - `var a=10;`
  - `var a="allo";`
  - `var a=10.3456;`

cours9p1

Les variables

- Les instructions se terminent par ';'
- Les variables sont identifiées par le mot réservé 'var' suivi par le nom de la variable. Nous appelons une telle instruction une déclaration de variable.
- Une valeur peut être chargée dans la variable au moment de sa déclaration:
  - `var a=5`
  - Un nom de variable débute par _ ou une lettre suivi par des lettres, des chiffres et quelques autres symboles.
- Les type de valeurs sont:
  - des entiers: `1787`
  - des nombres réels: `3.17635`
  - des booléens: `true, false`
  - des chaînes de caractères: `"ceci est une chaîne"`.

`var _abc` n'est pas la même que `var _Abc`

Si a="abc" et b="def" alors "a="+a+", b="+b vaut la chaîne: "a=abc, b=def"

Ce code javascript est exécuté au moment où la page est affichée dans le navigateur.

Écrit sur la page

```
<!DOCTYPE html>
<html>
   <head>
     <meta charset="utf-8">
     <link href="monstylesimple.css" rel="stylesheet" media="all"
type="text/css">
     <title>Tester Javascript (les variables)</title>
     <script>
         var a=2;
         var A=3;
         var b="allo";
         var B='salut';
         var c=3.73;
         var d=true;
         document.write("Les variables sont a="+a+", A="+A+",
b="+b+", B="+B+", c="+c+" et d="+d);
     </script>
 </head>
 <body>
   <h1>Tester javascript</h1>
 </body>
</html>
```

Placer le script dans le corps

cours9p2

Les variables sont a=2, A=3, b=allo, B=salut, c=3.73 et d=true

Tester les variables Javascript

```
<!DOCTYPE html>
<html>
   <head>
     <meta charset="utf-8">
     <link href="monstylesimple.css" rel="stylesheet"
media="all" type="text/css">
     <title>Tester Javascript (les variables)</title>
     <script>
         var a=2;
         var A=3;
         var b="allo";
         var B='salut';
         var c=3.73;
         var d=true;
         document.write("Les variables sont a="+a+", A="+A+",
b="+b+", B="+B+", c="+c+" et d="+d);
     </script>
 </head>
 <body>
   <h1>Tester les variables javascript</h1>
 </body>
</html>
```

Tester les variables javascript

Les variables sont a=2, A=3, b=allo, B=salut, c=3.73 et d=true

```
<!DOCTYPE html>
<html>
   <head>
     <meta charset="utf-8">
     <link href="monstylesimple.css" rel="stylesheet"
media="all" type="text/css">
     <title>Tester Javascript (les variables)</title>
   </head>
 <body>
   <h1>Tester les variables javascript</h1>
   <script>
         var a=2;
         var A=3;
         var b="allo";
         var B='salut';
         var c=3.73;
         var d=true;
         document.write("Les variables sont a="+a+", A="+A+",
b="+b+", B="+B+", c="+c+" et d="+d);
   </script>
 </body>
</html>
```

Javascript à partir d'un fichier

- Les programmes javascript peuvent être rangés dans un fichier avec extension `js` pour être associé à une page.
- La balise `<script>` peut faire appel à un script rangé dans un fichier:
  - `<script type="text/javascript" src="fichier.js"></script>`

Opérateurs javascript

- Opérateurs arithmétiques:
  - `=,+,-,*,/ : a=b,a+b, a-b, a*b, a/b`
  - `++,-- : a++` (i.e. `a=a+1`), `a--` (i.e. `a=a-1`)

Si a et b sont des chaînes de caractères alors a+b est leur concaténation.

- Opérateurs booléens:
  - `==, != : a==b` (i.e égal), `a!=b` (i.e. différent)
  - `>,<,>=,<= : a>b,a<b`(i.e plus grand),`a>=b,a<=b`
- Opérateurs logiques:
  - `&&, || , ! `: le ET, le OU et le NON logique, `a&&b, a||b, !a`

Lorsque `a` et `b` sont des booléens.

Opérateurs javascript (II)

- Opérateurs d'affectations:
  - `=` : le plus simple, `a=b`, place la valeur de `b` dans `a`.
  - `+=`, `-=`, `*=`, `/=` : `a+=b` (i.e. `a=a+b`), `a-=b` (i.e. `a=a-b`), `a*=b` (i.e. `a=a*b`), `a/=b` (i.e. `a=a/b`).
- Opérateurs conditionnels:
  - `?`: par exemple `(a<b)?10:20` retourne 10 si `a<b` et 20 sinon.
- Opérateur de type `typeof`:
  - `typeof(a)` : retourne `"number"`, `"string"`, `"boolean"`, `"object"`, `"function"`, `"undefined"`, etc… en fonction de type de valeur rangée dans `a`.

If…else…

- L'énoncé conditionnel le plus général est le `if…then`:
  - `if(cond){instruction}`: exécute `instruction` si `cond` est vraie,
  - `if(cond){inst1}else{inst2}`: exécute `inst1` si la condition `cond` est satisfaite et exécute `inst2` sinon.
  - `if(cond1){inst1}else if(cond2){inst2} else {inst3}` : exécute `inst1` si `cond1` sinon `inst2` si `cond2` et `inst3` sinon.

cours9p3

Exemple

```
<script>
var a="allo";
var x = 1;
var y = 2;
if (typeof(a)=="string"){
  x++;
  y*=x;
}else{
   y*=x;
}
document.write("Le résultat est y="+y);
</script>
```

cours9p4

Les instructions sont mises entre {..}

Et le résultat imprimé sur la page sera:

`Le résultat est y=4`

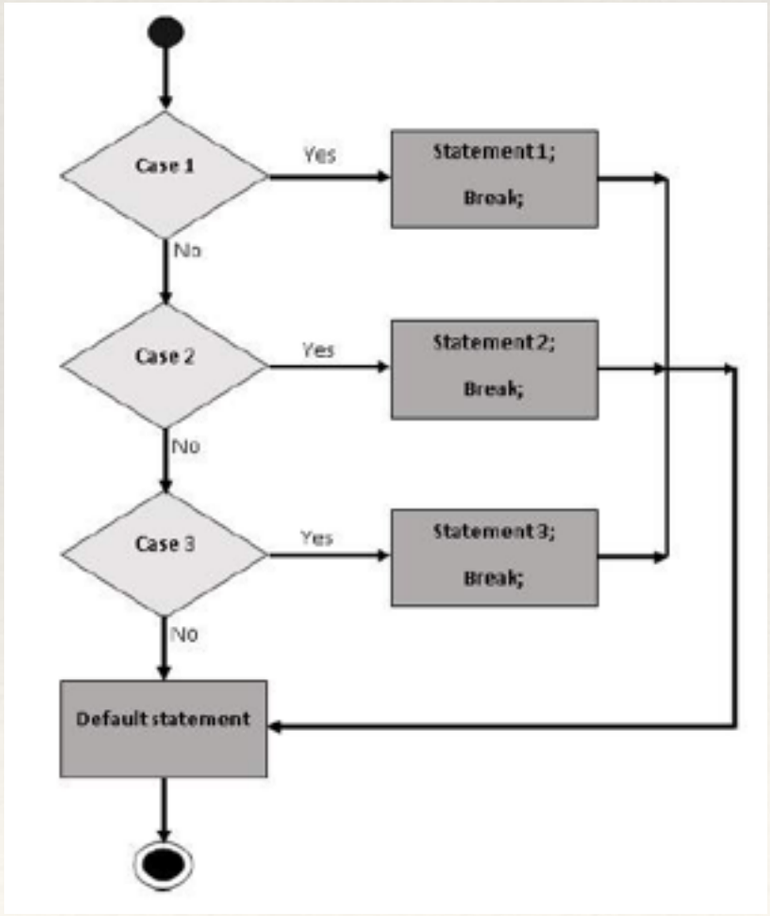
cours9p5

L'instruction `switch`

- L'instruction `switch` évalue une expression et exécute une séquence d'instructions en fonction de la valeur de l'expression. Chaque cas possible pour la séquence d'instructions est indiqué par une instruction `case`.

```
<script>
var x=2;
var y=3
switch(2*x+y){
case 0: document.write("c'est 0");
        break;
case 2,4,6: document.write("c'est paire plus grand que 0");
        break;
default: document.write("n'est pas paire entre {0,..6}");
        break;
}
</script>
```

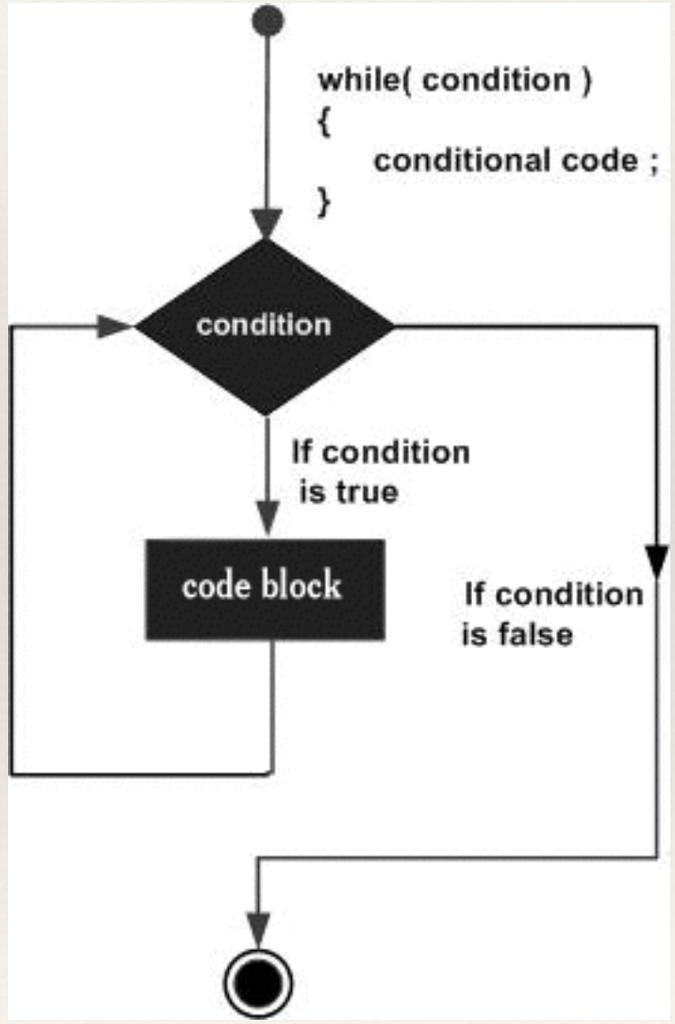
Indique la fin des instructions pour un `case`



cours9p6

Les boucles `while`

```
<script>
var x=1;
document.write("La boucle va débuter");
while (x<20){
  document.write("valeur de x courante:"+x+"<br/>");
  x*=2;
}
document.write("La boucle est terminée");
</script>
```

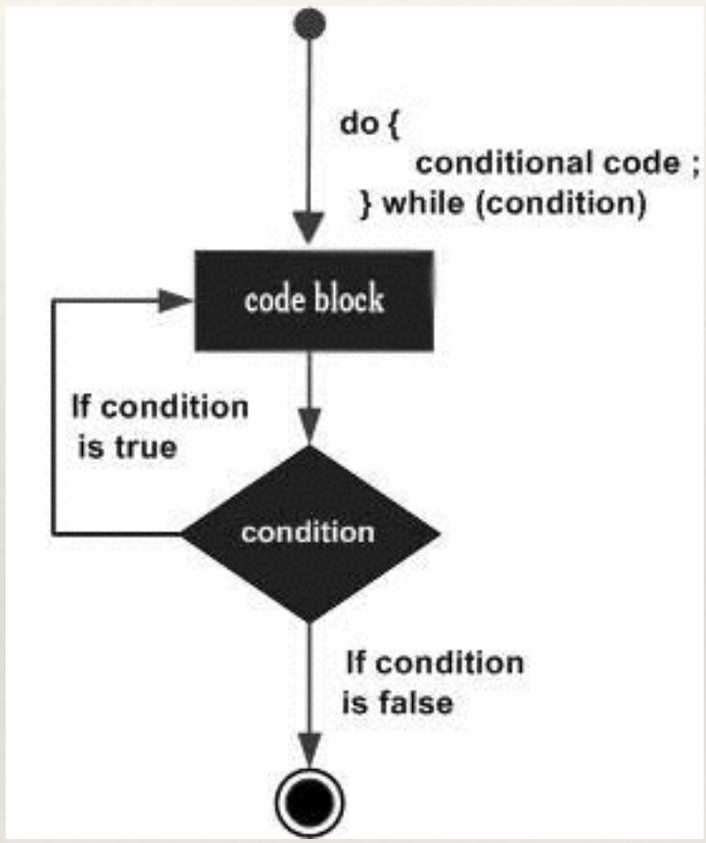


La boucle `while` est exécutée à chaque fois que la condition est satisfaite. La boucle se termine lorsque la condition est fausse.

cours9p7

Les boucles do . . while

```
<script>
var x=1;
document.write("La boucle débute");
do{
   x*=20;
   document.write("valeur de x courante:"+x+"<br/>");
 }while(x<20)
document.write("La boucle est terminée");
</script>
```



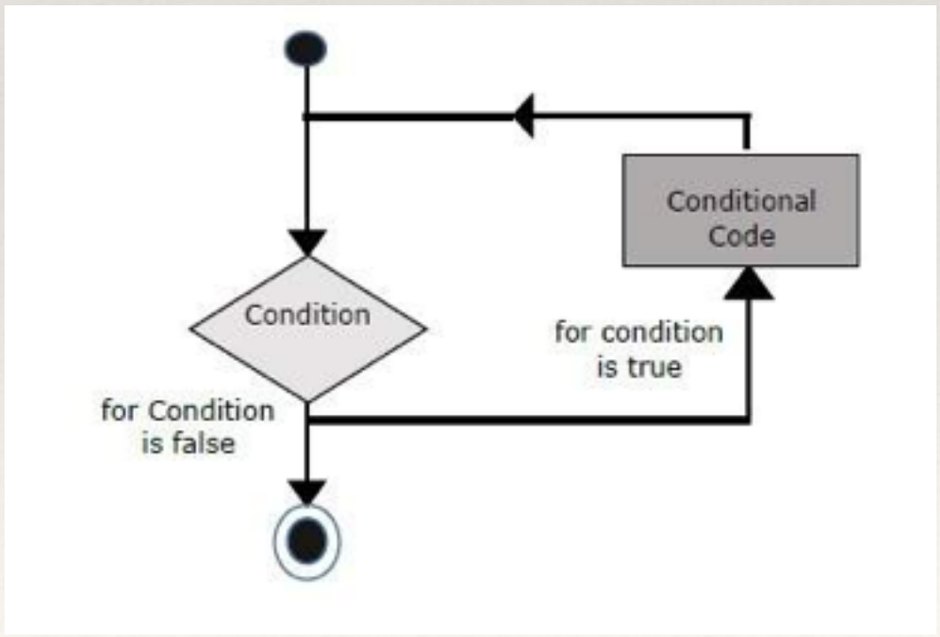
La boucle do exécute chaque instruction et vérifie ensuite si la condition est satisfaite. Si c'est le cas alors la boucle est terminée.

cours9p8

Les boucles `for`

```
<script>
var x;
document.write("La boucle débute");
for(x=0,x<20, x++){
   document.write("valeur de x courante:"+x+"<br/>");
 }
document.write("La boucle est terminée");
</script>
```

La boucle `for` est exécutée en partant avec `x=0` chaque fois que `x<20` et en ajoutant `1` à `x` à chaque itération.



cours9p9

Les boucles `for...in`

```
<script>
var x;
document.write("La boucle débute");
for(x in navigator){
   document.write(x+"<br/>");
 }
document.write("La boucle est terminée");
</script>
```

ceci est un objet!

nous verrons ce qu'est un objet plus tard.

Pour chaque élément tour à tour dans l'objet `navigator` (ici), la valeur de `x` devient cet élément et un tour de boucle est exécuté.

Pour terminer à propos des boucles

- L'instruction `break` peut être utilisée pour sortir d'une boucle, comme pour sortir du case dans un `switch`.
- L'instruction `continue` permet, lorsqu'elle est rencontrée, d'aller directement à la prochaine itération de la boucle.
- Des étiquettes peuvent être données à des boucles pour en sortir:

```
boucleext: while (i<20){
     boucleinterieur: while(j<30){
          if(cond1){break boucleext;}
          if(cond2){break boucleinterieur;}
     }
}
```

Sortira des 2 boucles

Sortira que de la boucle intérieure

Peut aussi être:
`continue boucleext`

Les fonctions

- Des bouts de programme qui sont utiles dans plusieurs contextes peuvent être définis comme des fonctions.
- Une fonction définie donc du code réutilisable à partir de n'importe où dans votre programme.
- Elles aident à écrire du code modulaire et à transformer de longs programmes en petites parties abstraites faciles à modifier.
    - Nous avons vu la fonction `write()` fournie par les objets nommée `document`. Nous y reviendront.
    - Nous pouvons définir nos propres fonctions...
- Les fonctions peuvent également retourner une valeur, la commande `return`, pour que le programme appelant puisse l'utiliser.

Fonctions: Exemple

- Les fonctions respectent cette forme:

```
function nom(arg1, arg2, ...){
   instructions;
}
```

- L'instruction

```
return expression;
```

dans les `instructions` d'une fonction retourne l'expression à l'instruction appelante.

N'est exécuté que lorsque `cmApc(variable)` est appelé

Une ligne de commentaires commencent par //

```
<script>
      function cmApc(lg){
        //convertir des centimetres
        //en pouces
          var conversCMPC = 0.39370;
          return lg*conversCMPC;
      }
      var long = 70;
      document.write(long+"cm vaut "
                  +cmApc(long)+"in.");
</script>
```

cours9p10

cours9p11

La visibilité des variables d'un script

```
<script>
     function cmApc(lg){
       //convertir des centimetres
       //en pouces
          var conversCMPC = 0.39370;
          document.write("dans la fonctions conversCMPC="+conversCMPC+".<br/>");
          document.write("long="+long);
          return lg*conversCMPC;
     }
     var long = 70;
     document.write(long+"cm vaut "
               +cmApc(long)+"in.<br/>");
     document.write("dans le programme principal conversCMPC n'est pas défini.<br/>");
</script>
```

Les événements

- L'interaction entre HTML et javascript est assurée par des événements qui se déroulent sur la page initiés par les actions de l'usager.
- Des événements sont les suivants:
  - La page se charge dans le navigateur,
  - L'utilisateur actionne un bouton,
  - L'utilisateur entre du texte,
  - Une touche est pressée,
  - La fenêtre du navigateur change de taille,
  - etc...

Les événements (II)

- Chaque élément HTML contient un ensemble d'événements qui peuvent déclencher du code javascript. Ces événements sont interceptés dans les attributs de balise:
- `<body>` : `onload`, `onunload`.
- `<form>`: `onchange`, `onsubmit`, `onreset`, `onselect`, `onblur`, `onfocus`.
- Au clavier: `onkeydown`, `onkeypress`, `onkeyup` pour la plupart des balises.
- `<balise>`: `onclick`, `ondbclick`, `onmousedown`, `onmousemove`, `onmouseout`, `onmouseover`, `onmouseup`. Fonctionne pour la plupart des balises (sauf `<head>`, `<html>`, `<meta>`).
- etc...

Exemple: lancer javascript par un bouton

cours9p12

```
<!DOCTYPE html>
<html>
      <head>
      <meta charset="utf-8">
      <link href="monstylesimple.css" rel="stylesheet" media="all" type="text/css">
      <title>Tester Javascript (les événements)</title>
      <script>
        function cmApc(lg){
            // convertir des centimetres en pouces
            var conversCMPC = 0.39370;
            return lg*conversCMPC;
        }
        function ExtraireEtConvertir(){
            var mssg = "La longueur en pouces est:";
            alert("Hummm, comment aller chercher le nombre entré par l'usager?");
        }
        </script>
      </head>
      <body>
            <h1>Conversion de centimètres en pouces</h1>
                  Entrez votre longueur:
                  <input type="number" id="longueur" value="0" min="0" max="10000">
                  cm.<br>
                  <button type="button" onclick="ExtraireEtConvertir()">Convertir</button>
        </body>
</html>
```

Ceci est
un dialogue d'alerte qui informe
l'usager et disparaît en appuyant sur
OK.

Les objets vont
nous aider à faire ça!

cours9p13

Une autre façon de faire

```
<!DOCTYPE html>
<html>
     <head>
     <meta charset="utf-8">
     <link href="monstylesimple.css" rel="stylesheet" media="all" type="text/css">
     <title>Tester Javascript (les événements)</title>
     <script>
        function cmApc(lg){
              // convertir des centimetres en pouces
               var conversCMPC = 0.39370;
               return lg*conversCMPC;
        }
        function ExtraireEtConvertir(){
               var chainelgr = prompt("Entrez la longueur en cm à convertir:","0");
               var longueur = Number(chainelgr);
               if(!isNaN(longueur)){
                  document.write("Réponse: "+longueur+"cm vaut "+cmApc(longueur)+"in.<br/>");
               }else{
                  alert("Vous n'avez pas entré un nombre valide, essayez encore");
               }
        }
     </script>
     </head>
     <body>
               <h1>Conversion de centimètres en pouces</h1>
                 <button type="button" onclick="ExtraireEtConvertir()">Convertir</button>
        </body>
</html>
```

Les objets

- Javascript est un langage orienté objet:
  - **Encapsulation:** Il est possible de définir des éléments qui regroupent plusieurs propriétés communes aux objets d'une certaine classe. Ces caractéristiques peuvent être des propriétés et des méthodes (des fonctions). L'encapsulation permet de définir des objets d'un certain type incluant leurs fonctionnalités.
  - **Agrégation:** Des objets peuvent contenir d'autres objets.
  - **Héritage:** Des objets qui sont des spécialités d'autres objets héritent de leurs données encapsulées.
  - **Polymorphisme:** Les méthodes (ou fonctions) peuvent fonctionner sur des données de plusieurs types.

Les objets (II)

- Nous avons vu des objets javascript sans les mentionner. Par exemple:
  - `document`:
    - `document` est un objet qui encapsule les données relatives au document dans lequel le script java est exécuté. Cet objet est créé au début de l'exécution du script de la page.
  - `document.write(.)` est une méthode associée au document qui imprime l'argument sur le document lorsqu'elle est appelée par le script.
  - `document.URL` est une propriété du document qui contient l'URL du document sous la forme d'une chaîne.
  - Il y beaucoup beaucoup de propriétés et méthodes pour les objets de la classe des documents. Ils sont accessibles par n'importe quel script qui a accès à l'objet document.

Créer des objets en javascript

- Les tableaux sont des objets javascript qui permettent de ranger plusieurs éléments, chacun placé dans une case avec son numéro de case.
- En javascript, un objet de nom `monObjet` peut être créé avec l'instruction new:
  - `var monObjet = new Object();`
- Par exemple, un objet auto pourrait être défini avec une marque, un modèle, une année:
  - `var auto = new  Object();`
  - `auto.marque = "Ford";`
  - `auto.modele = "modèleT";`
  - `auto.annee = 1912;`

Fait référence à l'objet créé par l'appel à `new Automobiles(.,.,.)`

Pour définir le type d'objets `Automobiles` dont `auto` fait partie:

```
function Automobiles(laMarque, leModele, lAnnée){
  this.marque = laMarque;
  this.modele = leModele;
  this.annee = lAnnee;
}
```

est appelé un **prototype** en javascript

et l'objet `auto` (et `auto2`) peut être créée par:

```
var auto=new Automobiles("Ford", "modèleT", 1912);
var auto2 = new Automobiles("Ferarri", "512BB", 1976);
```

Les types d'objets prédéfinis

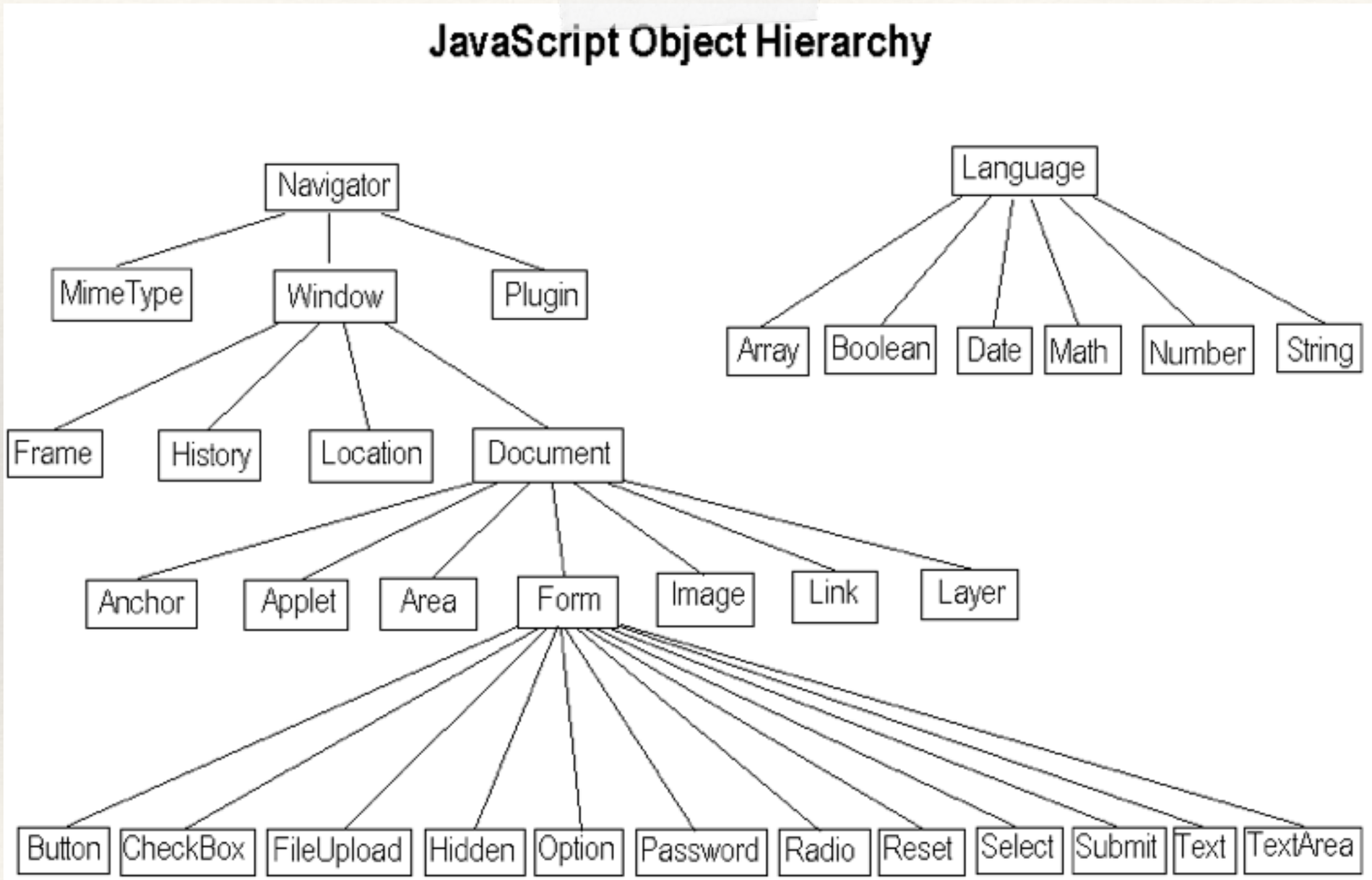
- Plusieurs types d'objets sont prédéfinis en javascript:
  - Le type d'objet `Array` permet de ranger plusieurs valeurs, chacun dans une case numérotée:
    - `var table = new Array(1,"ab", "cd",4);` ou
      - création de la table [1,"ab","cd",4] dans `table`.
    - `var table = new Array(4);`
      - création d'une table vide de 4 cases dans `table`.

Plusieurs méthodes sont disponibles pour les objets `Array`: `sort()`, `indexOf(val)`, `push(elem)`, `pop()`,...

les objets `Array ont la propriété: length` qui retourne le nombre d'éléments dans le tableau

  - Les types `Number`, `Math`, `Boolean`, `String`, `Date`, ... sont d'autres types d'objet prédéfinis...

La hiérarchie des objets prédéfinis



Accéder aux valeurs entrées par l'usager

- Javascript a accès aux valeurs entrées par l'usager sur la page web en ajoutant aux paramètres des balises `<input>`, des identificateurs qui dénoteront ces valeurs.

```
<input type="text" id="lgr" value="0">
```

- La fonction pourra ensuite avoir accès aux valeurs entrées en invoquant la méthode `getElementById(id)` que les objets documents fournissent:

```
var l = document.getElementById("lgr").value;
```

cour9p14

Retour sur la conversion

```
<!DOCTYPE html>
<html>
     <head>
     <meta charset="utf-8">
     <link href="monstylesimple.css" rel="stylesheet" media="all" type="text/css">
     <title>Tester Javascript (les événements)</title>
     <script>
       function cmApc(lg){
          // convertir des centimetres en pouces
           var conversCMPC = 0.39370;
           return lg*conversCMPC;
       }
       function ExtraireEtConvertir(){
           var longueur = document.getElementById("lgr").value;
           if(!isNaN(longueur)){
                document.write("Réponse: "+longueur+"cm vaut "+cmApc(longueur)+"in.<br/>");
           }else{
                alert("Vous n'avez pas entré un nombre valide, essayez encore");
           }
       }
     </script>
     </head>
     <body>
            <h1>Conversion de centimètres en pouces</h1>
                 Entrez votre longueur:
                 <input  type="text" id="lgr" value="0" size="10">cm.<br>
                 <button type="button" onclick="ExtraireEtConvertir()">Convertir</button>
     </body>
</html>
```

Ceci efface la page intiale :-(

cours9p15

Retour sur la conversion (II)

```
<script>
   function cmApc(lg){
      // convertir des centimetres en pouces
       var conversCMPC = 0.39370;
       return lg*conversCMPC;
   }
   function ExtraireEtConvertir(){
       var longueur = document.getElementById("lgr").value;
       if(!isNaN(longueur)){
            document.getElementById("sortie").innerHTML="Réponse: "+longueur+"cm vaut "
                                                          +cmApc(longueur)+"in.<br/>";
       }else{
            alert("Vous n'avez pas entré un nombre valide, essayez encore");
       }
  }
</script>
</head>
<body>
       <h1>Conversion de centimètres en pouces</h1>
            Entrez votre longueur:
            <input  type="text" id="lgr" value="0" size="10">cm.<br>
            <button type="button" onclick="ExtraireEtConvertir()">Convertir</button>
            <p id="sortie"></p>
</body>
```

Ce qui est entre la balise ouvrante avec identificateur "sortie" et la balise fermante correspondante

la résultat sera imprimé ici

Mettre tout ça ensemble

- Voici un petit projet:
  - L'usager sélectionne deux octets de 8 bits,
  - le script représente l'octet qui résulte en l'addition des deux octets donnés.
  - Les octets sont représentés par des boutons radios,
    - le bouton est non sélectionné: la valeur `0` est rangée,
    - le bouton est sélectionné: la valeur `1` est rangée.
- Le résultat est donné sous forme d'octet représenté de la même façon par des boutons radios.
- L'application donnera également les valeurs correspondantes en notation décimale.

cours9p16

Le code HTML

```
<div  onclick="addEtMontre()">
<table>
<tr>
<th> </th>
<th>7</th><th>6</th><th>5</th>
<th>4</th><th>3</th><th>2</th><th>1</th><th>0</th>
</tr>
<tr>
<td> </td>
<td><input type="checkbox" id="a7"></button></td>
<td><input type="checkbox" id="a6"></button></td>
<td><input type="checkbox" id="a5"></button></td>
<td><input type="checkbox" id="a4"></button></td>
<td><input type="checkbox" id="a3"></button></td>
<td><input type="checkbox" id="a2"></button></td>
<td><input type="checkbox" id="a1"></button></td>
<td><input type="checkbox" id="a0"></button></td>
<td id="vala"> </td>
</tr>
<tr>
<td>+</td><td> </td><td> </td><td> </td><td> </td>
<td> </td><td> </td><td> </td><td> </td>
</tr>
<tr>
<td></td>
<td><input type="checkbox" id="b7"></button></td>
<td><input type="checkbox" id="b6"></button></td>
<td><input type="checkbox" id="b5"></button></td>
<td><input type="checkbox" id="b4"></button></td>
<td><input type="checkbox" id="b3"></button></td>
<td><input type="checkbox" id="b2"></button></td>
<td><input type="checkbox" id="b1"></button></td>
<td><input type="checkbox" id="b0"></button></td>
<td id="valb"> </td>
</tr>
```

```
<tr>
<td> </td>
<td><hr></td><td><hr></td><td><hr></td><td><hr></td>
<td><hr></td><td><hr></td><td><hr></td><td><hr></td>
</tr>
<tr>
<td> </td>
<td><input type="checkbox" disabled="disabled" id="r7"></td>
<td><input type="checkbox" disabled="disabled" id="r6"></td>
<td><input type="checkbox" disabled="disabled" id="r5"></td>
<td><input type="checkbox" disabled="disabled" id="r4"></td>
<td><input type="checkbox" disabled="disabled" id="r3"></td>
<td><input type="checkbox" disabled="disabled" id="r2"></td>
<td><input type="checkbox" disabled="disabled" id="r1"></td>
<td><input type="checkbox" disabled="disabled" id="r0"></td>
<td id="valr"> </td>
</tr>
</table>
</div>
<p id="sortie"></p>
```

cours9p16

Fonction pour convertir binaire en décimal

```
function binAdec(ids){
    var z=1;
    var val=0;
    for(var x=0;x<8;x++){
        if(document.getElementById(ids+x).checked){
            val += z;
        }
        z *= 2;
    }
    return val;
}
```

cours9p16

Fonction pour additionner et montrer

```
function addEtMontre(){
 var a=0;var b=0; var z=false;var r=0;
 var va = binAdec("a");var vb = binAdec("b");
 var vr = binAdec("r"); var rr = va+vb;
 for(var x=0; x<8; x++){
    (document.getElementById("a"+x).checked)?a=1:a=0;
    (document.getElementById("b"+x).checked)?b=1:b=0;
     switch(a+b+r){
          case 0: z=false; r=0; break;
          case 1: z=true; r=0; break;
          case 2: z=false; r=1; break;
          case 3: z=true; r=1; break;
     }
     document.getElementById("r"+x).checked = z;
  }
  document.getElementById("vala").innerHTML = "="+va;
  document.getElementById("valb").innerHTML = "="+vb;
  document.getElementById("valr").innerHTML = "="+vr;
  document.getElementById("sortie").innerHTML = "Le résultat en décimal serait: "+va+"+"+vb+"="+rr;
}
```