Javascript

Eric SULTAN

0212v3

Contenu du cours

- Introduction
- Base du langage
- Structures de contrôle
- Types / Classes
- Navigateur
- DOM
- Evènements
- jQuery

Javascript

INTRODUCTION

Introduction

- JavaScript est un langage de programmation de scripts principalement utilisé dans les pages web interactives (comme par exemple, la validation de données d'un formulaire).
- Il est formé de trois composants:
  - **ECMAScript**, qui est défini dans l'édition ECMA-262, et qui fournit les fonctionnalités centrales
  - **DOM (Document Object Model)** qui fournit les fonctionnalités pour interagir avec le contenu d'une page web
  - **BOM (Browser Object Model)** qui fournit les fonctionnalités pour interagir avec le navigateur

Javascript dans HTML

- L'élément HTML script permet d'intégrer du code Javascript dans une page.
- Les attributs de cet élément sont :
  - **type** : indique le type de contenu (appelé aussi type MIME).
    - La valeur est typiquement "text/javascript".
  - **charset** (optionnel) : indique le jeu de caractères utilisé.
  - **defer** (optionnel) : indique si l'exécution du script doit être décalée.
  - **src** (optionnel) : indique que le code se situe dans un fichier externe

Inline Code

- Il suffit d'utiliser uniquement l'attribut type et de placer le code au cœur de l'élément script.

```
<script type="text/javascript">
        function sayHi() {
                alert("Hi!");
        }
</script>
```

External Files

- Il suffit d'utiliser uniquement l'attribut type avec l'attribut src.

```
<head>
<title> Example </title>
<script type="text/javascript" src="example1.js"> </script>
<script type="text/javascript" src="example2.js"> </script>
```

Inline code et symboles spéciaux

- Le code suivant n'est pas correct en XHTML :

```
<script type="text/javascript">
        function test(a,b) {
            if (a < b) alert("premier parametre plus petit");
        }
</script>
```

  - Première solution : replacer < par <
  - Deuxième solution :

```
<script type="text/javascript">
// <![CDATA
        function test(a,b) {
            if (a < b) alert("premier parametre plus petit");
        }
// ]]>
</script>
```

Inline code ou External Files ?

- Il est préférable d'utiliser des fichiers externes pour des raisons de :
  - maintenabilité : le code JavaScript peut être rassemble dans un unique répertoire
  - caching : un fichier js partagé par deux pages ne sera téléchargé qu'une seule fois
  - lisibilité : pas besoin d'astuces telles que // ... [CDATA qui polluent la page

L'element noscript permet d'acher un contenu lorsque le navigateur ne supporte pas les scripts ou que le support a ete desactive. Par exemple :

```
<body>
        <noscript>
                <p> Cette page necessite un support pour JavaScript </p>
        </noscript>
```

JavaScript

BASES DU LANGAGE

Identificateurs et commentaires

- Les identificateurs
  - sont sensibles a la casse: **test** n'est pas **Test**
  - sont formes par convention en utilisant le style camel case comme dans *sommeNotes*

```
Les commentaires
// single line comment
/*
 * multi-line
 * comment
 */
```

Identificateurs et commentaires

- Les instructions
  - se terminent par un point-virgule
  - nécessitent des accolades lorsqu'elles forment un bloc

```
var sum = a + b;
...
if (sum > 0) alert("positif");
...
if (a < b) {
        var tmp = a;
        a = b;
        b = tmp;
        }
```

Variables

- Déclarer une variable nécessite l'utilisation du mot-clé **var**.

```
var sum;
...
var v = 100;
...
v = "coucou"; // autorise mais deconseille
...
var x = 10, y = 20, z = false; // multiple declaration
```

Portée locale :

```
function test() {
        var message = "hi";
}
test();
alert(message); // undefined
```

Portée globale :

```
function test() {
        message = "hi";
}
test();
alert(message); // « hi »
```

Valeurs undefined et null

- La valeur undefined fait référence a une variable non déclarée ou déclarée mais pas initialisée, null fait référence à une variable censée référencer un objet mais non encore disponible.

```
var message = "coucou";
alert (typeof message); // "string"
var m1;
alert(m1); // "undefined"
alert(m2); // erreur
alert(typeof m1); // "undefined"
alert(typeof m2); // undefined
...
var car = null;
alert(typeof car); // "object"
```

Boolean et Number

- Le type Boolean a deux valeurs true et false.
- Le type Number sert a représenter des valeurs entières et des valeurs flottantes (réelles).

```
var x = 55;
var y = 070; //octal pour 56 (en base 10)
var z = 0xA, //hexadecimal pour 10
var f1 = 10.2;
var f2 = 3.12e7; // represente 31 200 000
var f3 = 3e-10; // represente 0,0000000003
```

Les valeurs particulières du type Number sont :
- Number.MIN VALUE, Number.MAX VALUE
- Number.NEGATIVE INFINITY, Number.POSITIVE INFINITY
- NaN (Not a Number)

Conversions numeriques

- Relatifs aux valeurs et conversions des nombres, on trouve 4 fonctions :
  - isNaN détermine si un paramètre donne n'est pas un nombre
  - Number effectue une conversion
  - parseInt effectue une conversion en valeur entière
  - parseFloat effectue une conversion en valeur réelle

```
alert(isNaN(10)); // false
alert(isNaN("10")); // false - peut être convertie
alert(isNaN("blue")); // true - ne peut être convertie
var num1 = Number("hello world"); // NaN
var num2 = Number("00001"); // 1
var num3 = Number(true); // 1
var num3 = parseInt(""); // NaN
var num4 = parseInt(22.5); // 22
var num5 = parseInt("70",10); // 70 - la base 10 est spécifiée
var num6 = parseFloat("22.5"); // 22.5
```

String

- On utilise les quottes simples (apostrophes) ou doubles (guillemets) pour définir des valeurs chaînes de caractères.
- L'attribut **length** permet de déterminer la longueur d'une chaîne.

```
var nom = "Wilde";
var prenom = 'Oscar';
alert(prenom.length); // 6
var message = "toto a dit \"je suis malade\".";
```

- Il est possible de **transtyper** une valeur en chaîne avec String().

```
var v1 = 10; alert(String(v1)); // "10"
var v2 = true; alert(String(v2)); // "true"
var v3 = null; alert(String(v3)); // "null"
var v4; alert(String(v4)); // "undefined"
```

String

- Il existe de nombreuses fonctions sur les chaînes de caractères.

```
var s = "hello world";
alert(s.length); // 11
alert(s.charAt(1)); // "e"
alert(s.charCodeAt(1)); // 101
alert(s.slice(3)); // "lo world"
alert(s.slice(-3)); // "rld"
alert(s.substring(3,7)); // "lo w"
alert(s.indexOf("o")); // 4
alert(s.lastIndexOf("o")); // 7
alert(s.toUpperCase()); // HELLO WORLD
alert(s + " !"); // hellow world !
```

Math

- Il s'agit d'un objet définissant de nombreuses constantes et fonctions mathématiques.

```
alert(Math.E); // la valeur de e
alert(Math.PI); // la valeur de pi
alert(Math.min(5,12)); // 5
alert(Math.max(23,5,7,130,12)); // 130
alert(Math.ceil(25.3)); // 26
alert(Math.floor(25.8)); //25
alert(Math.random()); // valeur aleatoire entre 0 et 1
var n = Math.floor(Math.random()*nb + min);
alert(n); // valeur aleatoire entre min et min+nb (exclus)
```

- D'autres fonctions :
  - Math.abs(x) Math.exp(x) Math.log(x)
  - Math.pow(x,y) Math.sqrt(x)
  - Math.sin(x) Math.cos(x) Math.tan(x)

Opérateurs

- Typiquement, ceux de C, C++ et java:
  - incrémentation/décrémentation (++, --)
  - arithmétiques (+, -, =, *, /, %)
  - relationnels (>,<,>=,<=,==,! =) et (===,! ==)
  - logique (!,&&,||)
  - affectation (=,+=,-=,*=,/=,%=)
  - bit a bit (&,|,<<,>>, ...)

```
var age = 10;
age++;
alert(age); // 11
alert(age > 10 && age < 20); // true
alert(26 % 5); // 1
age*=2;
alert(age); // 22
```

JavaScript

STRUCTURES DE CONTRÔLE

Structures de contrôle

- Elles sont très proches de celles de langages tels que C, C++ et Java. Pour rappel, les structures de contrôle sont de trois types :
  - **Séquence** : exécution séquentielle d'une suite d'instructions séparées par un point-virgule
  - **Alternative** : structure permettant un choix entre divers blocs d'instructions suivant le résultat d'un test logique
  - **Boucle** : structure itérative permettant de répéter plusieurs fois le même bloc d'instructions tant qu'une condition de sortie n'est pas avérée

Il ne faut pas confondre x == y (test d'égalité) avec x = y (affectation).

Alternative

- L'instruction **if** sans partie else :

```
if (condition) instruction;
```

```
if (condition) {
        instruction1;
        instruction2;
        ...
}
```

```
if (x >= 0) alert("valeur positive ou nulle");
...
if (note > 12 && note <= 14) {
                alert("bravo");
                mention="bien";
}
```

Alternative

- L'instruction **if...else** :

```
if (condition)
                instruction1;
else
                instruction2;
```

```
if (condition1) {
                instructions1;
} else if (condition2) {
                instructions2;
} else {
                instructions3;
}
```

```
if (rank == 1)
                medaille="or";
else if (rank == 2)
                medaille="argent";
else if (rank == 3)
                medaille="bronze";
```

Alternative

- L'operateur ternaire ?: permet de remplacer une instruction if...else simple.
  - Sa syntaxe (lorsqu'utilisée pour donner une valeur à une variable) est :
    - variable = condition ? expressionIf : expressionElse;
  - Elle est équivalente à :
    - if (condition) variable=expressionIf;
    - else variable=expressionElse;

```
var civilite = (sexe == "F") ? "Madame" : "Monsieur";
```

Boucle

- L'instruction **while** :

```
while (condition) instruction;
```

```
while (condition) {
        instruction1;
        instruction2;
        ...
}
```

```
var num = 1;
while (num <= 5) {
        alert(num);
        num++;
}
```

Boucle

- L'instruction **for**:

```
for (instructionInit; condition; instructionIter)
        instruction;
```

```
for (instructionInit; condition; instructionIter) {
    instruction1;
    instruction2;
    ...
}
```

```
for (var num = 1; num <= 5; num++)
        alert(num);
```

Boucle

- L'instruction **do...while** :

```
do {
    instruction1;
    instruction2;
    ...
}
while (condition);
```

- L'instruction **for-in** pour les objets :

```
for (var prop in window)
        document.writeln(prop);
```

Boucle

- Certaines instructions permettent un contrôle supplémentaire sur les boucles :
  - **break** permet de quitter la boucle courante
  - **continue** permet de terminer l'itération en cours de la boucle courante

```
for (var i=0; i<5; i++) {
        for (var j=0; j<5; j++) {
                if ((i+j)%3 == 0)
                        continue;
                for (var k=0; k<5; k++) {
                        if ((i+j+k)%3 == 0)
                                break;
                        alert(i + " " + j + " " + k);
                }
        }
}
```

JavaScript

TYPES ET CLASSES

Object

- Utilise pour stocker des données. Création d'une variable (de type Object) avec new Object() ou de manière littérale (énumération entre accolades).

```
var person = new Object();
person.name = "Julien";
person.age = 23;
var person = {
        name : "Julien",
        age : 23
}
```

```
Il est possible d'utiliser les crochets pour accéder à un champ.
        alert(person.name);
        alert(person["name"]);
        var field="name"; alert(person[field]);
```

Array

- Les tableaux peuvent contenir des données de nature différente.

```
var colors = new Array(); // tableau vide
var colors2 = new Array(20); // tableau avec 20 cases
var colors3 = new Array("red","blue","green"); // 3 cases
var colors4 = ["red","blue","green"]; // notation littérale
```

```
Le champ length indique la taille d'un tableau.
        var colors = ["red","blue","green"];
        alert(colors.length); // 3
        colors[colors.length]="black"; // nouvelle couleur
        colors[99]="pink";
        alert(colors.length); // 100
        alert(colors[50]): // undefined
        colors.length=10; // plus que 10 cases
```

Array

- De nombreuses méthodes existent sur les tableaux.

```
var colors = ["red","blue","green"];
alert(colors); // red,blue,green
alert(colors.join(";")); // red;blue;green
colors.push("black");
alert(colors); // red,blue,green,black
var item = colors.pop();
alert(item + " " + colors); // black red,blue,green
var item2= colors.shift();
alert(item2 + " " + colors); // red blue,green
```

Il existe d'autres méthodes telles que **concat**, **slice** et **splice**.

Réordonner les tableaux

- On peut utiliser reverse et sort.

```
var values = [0, 1, 2, 3, 4];
alert(values.reverse()); // 4,3,2,1,0
values = [0, 1, 5, 10, 15];
alert(values.sort()); // 0,1,10,15,5
function compare(value1, value2) {
        ...
}
values = [0, 1, 5, 10, 15];
alert(values.sort(compare)); // 0,1,5,10,15
```

Fonctions

- La syntaxe pour définir une fonction est :

```
function name(arg0, arg1, ..., argN) {
    statements
}
```

```
function sayHi(name) {
    alert("Hello " + name);
}
sayHi("Nicolas");
```

```
function sum(num1, num2) {
    return num1+num2;
}
alert(sum(5,10)); //15
```

Une fonction peut retourner un résultat avec l'instruction return même si rien ne l'indique au niveau de la signature (en-tête) de la fonction.

Arguments

- Il existe toujours un tableau arguments implicite lors de l'appel a une fonction.

```
function f() {
    alert(arguments.length);
    for (var i=0; i<arguments.length; i++)
        alert(arguments[i]);
}
f("Nicolas"); // affiche 1 Nicolas
f("Nicolas",25); // affiche 2 Nicolas 25
```

La possibilité de faire appel a une fonction avec un nombre variable de paramètres pallie l'impossibilité de surcharge (overloading).

Fonctions comme objets

- Il y a deux manières (sensiblement équivalentes) de définir une fonction.

```
function sum(n1, n2) {
        return n1+n2;
}
alert(sum(5,10)); //15
```

```
var sum = function (n1, n2) {
        return n1+n2;
};
alert(sum(5,10)); //15
```

Il est possible de référencer une fonction par deux variables différentes ou de passer une fonction comme paramètre à une autre fonction.

Window

- L'objet **window** représente la fenêtre du navigateur.

```
window.moveTo(0,0);
window.resizeTo(800,800);
var wroxWindow = window.open("http://www.wrox.com","wrox");
if (wroxWindow == null)
	alert("fenetre bloquee");
else {
	...
	wroxWindow.close();
}
```

Le navigateur peut être configuré de manière a empêcher de modifier son emplacement, de modifier sa taille ou encore d'afficher une fenêtre pop-up.

Timer

- Il est possible de programmer l'exécution d'une méthode à un instant donné ou à des temps réguliers grâce a **setTimeout** et **setInterval**.

```
function helloWorld() {
	alert("hello world");
}
var id = setTimeout(helloWorld, 1000);
var num=0, max=4, intervalId = setInterval(incrementNumber,500);
function incrementNumber() {
	if (++num == max) clearInterval(intervalId);
	else alert(num);
}
```

Il est possible d'annuler avec clearTimeout et clearInterval.

Boites de dialogues

- Des boites de dialogues peuvent être ouvertes en utilisant les méthodes **alert**, **confirm** et **prompt**.

```
if (confirm("Are you sure?"))
	alert("I am glad");
else
	alert("I am sorry");

var name = prompt("What is your name?","Toto");
if (name != null)
	alert("Welcome " + name);
```

Il existe deux autres méthodes asynchrones **print** et **find**.

Objets du BOM (Browser Object Model)

- En plus de **window**, il est possible d'utiliser les objets **location**, **navigator**, **screen** et **history**.

```
location.href="http://www.wrox.com";
location.reload();
location.port=8080; // change le port
alert(navigator.appName); // nom du navigateur
alert(navigator.javaEnabled);
alert(screen.colorDepth);
alert(screen.width);
window.resizeTo(screen.availWidth,screen.availHeight);
history.go(2); // go forward 2 pages
history.back(); // go back one page
if (history.length == 0) alert("this is the first page");
```

JavaScript

DOM
DOCUMENT OBJECT MODEL

DOM

- DOM (Document Object Model) est une API pour manipuler les documents HTML et XML.
- L'objet document représente la page HTML chargée par le navigateur.

```
var html = document.documentElement;
alert (html == document.firstChild); // true
var body = document.body;
document.title="nouveau titre";
var url = document.URL;
var domain = document.domain;
alert(url + " " + domain);
```

Méthode getElementById()

- Cette méthode prend comme argument l'id d'un élément.

```
// dom.html
    <head>
        ...
    </head>
    <body>
        <p id="p1"> An ordered list: </p>
        ...
        <script type="text/javascript" src="dom.js"> </script>
    </body>

// dom.js
    var p = document.getElementById("p1");
    p.style.color="blue";
```

Méthode getElementsByTagName()

- Cette méthode prend comme argument le nom de balise des éléments à récupérer.

```
// dom.html
    <body>
        <ol>
            <li>XHTML</li>
            <li>CSS</li>
            ...
        </ol>
        <script type="text/javascript" src="dom.js"> </script>
    </body>

// dom.js
    var list = document.getElementsByTagName("li");
    for (var i=0; i< list.length; i++)
        list[i].style.color="green";
```

Propriété innerHTML

- En utilisant cette propriété, il est possible de modifier le contenu d'un élément.

```
var div1 = document.getElementById("d1");
var div2 = document.getElementById("d2");
var div3 = document.getElementById("d3");
div2.innerHTML=div1.innerHTML;
div3.innerHTML="je suis <strong> content </strong>";
```

La propriété **innerText** (*textContent* pour Firefox) est similaire a **innerHTML**, mais ne traite que du texte simple.

Créer des éléments HTML

- Il existe de nombreuses méthodes DOM pour créer dynamiquement des éléments.

```
var table = document.createElement("table");
for (var i=1; i<=10; i++) {
	var row = document.createElement("tr");
	for (var j=1; j<=10; j++) {
		var cell = document.createElement("td");
		cell.appendChild(document.createTextNode(i*j));
		row.appendChild(cell);
	}
	table.appendChild(row);
}
document.getElementById("d1").appendChild(table);
```

Modifier le style dynamiquement

- Tout élément HTML dispose d'un attribut **style** en JavaScript. Les noms des propriétés CSS doivent être convertis en camel case. Par exemple :

Propriété CSS	Propriété JavaScript
background-image	style.backgroundImage
color	style.color
font-family	style.fontFamily

```
var myDiv = document.createElement(" div ");
myDiv.style.backgroundColor="red";
myDiv.style.width="100px";
myDiv.style.border="1px solid border";
```

La propriété float correspond a un mot réservé de Javascript. Il faut alors utiliser cssFloat (ou styleFloat pour IE).

JavaScript

EVÈNEMENTS

Evènements

- Un évènement est provoqué par une action de l'utilisateur ou du navigateur lui-même.
- Les évènements ont des noms tels que **click**, **load** et **mouseover**.
- Une fonction appelée en réponse a un évènement se nomme un écouteur (**event handler ou event listener**).
  - Souvent, leur nom commence par on comme par exemple **onclick** ou **onload**.
- Associer des écouteurs aux évènements possibles peut se faire de trois manières différentes:
  - HTML
  - DOM Level 0
  - DOM Level 2 (pas présenté ici)

HTML Event Handlers

- On utilise des attributs HTML pour déclarer les écouteurs.
- La valeur de ces attributs est le code JavaScript à exécuter lorsque l'évènement est produit.

```
<input type="button" value="but1" onclick="alert('clicked')" />
<script type="text/javascript">
        function showMessage() {
                alert("hello world");
        }
</script>
<input type="button" value="but2" onclick="showMessage()" />
```

Il faut parfois échapper des caractères. Par exemple :

```
onclick="alert(&amp;quot;clicked&amp;quot;)"
```

Dom Level 0 Event Handlers

- On utilise les propriétés des éléments pour leur associer des écouteurs.

```
// dom0.html
    <input type="button" id="but1" value="bouton 1" />

// dom0.js
    function but1Listener() {
        alert("but1 cliqued");
    }
    var but1 = document.getElementById("but1");
    but1.onclick=but1Listener;
```

Pour éliminer l'écouteur, on place la valeur null. Par exemple :

```
    but1.onclick=null;
```

L'objet **event**

- Quand un évènement se produit, toutes les informations le concernant sont enregistrées dans un objet appelé **event**.
  - Il est possible de récupérer cet objet sous forme de paramètre d'une fonction écouteur.

```
function but3Listener(event) {
    switch(event.type) {
        case "click" :
            alert("clicked at " + event.clientX + " " + event.clientY);
            break;
        case "mouseover" :
            this.style.backgroundColor="red"; break;
        case "mouseout" :
            this.style.backgroundColor=""; break;
    }
}
but3.onclick=but3Listener;
but3.onmouseover=but3Listener;
but3.onmouseout=but3Listener;
```

Evènements souris

- Ce sont :

click	dblclick
mousedown	mouseup
mouseover	mouseout
mousemove	

- Les propriétés utiles et accessibles à partir de l'objet **event** sont :

clientX	clientY
screenX	screenY
shiftkey	ctrlKey
altKey	metaKey

Evènements clavier

- Ce sont :

keydown	keyup
keypress	

- Les propriétés utiles et accessibles à partir de l'objet **event** sont :

shiftkey	ctrlKey
altKey	metaKey
keyCode	

- Exemples de valeurs pour **keyCode**:
  - 40 pour Down Arrow
  - 65 pour A
  - 112 pour F1

JavaScript

JQUERY

Bibliothèque jQuery

- Librairie Javascript developpee depuis 2006
- Avantages :
  - interface simple et puissante pour écrire du code
  - aplanit les différences entre navigateurs
  - beaucoup de ressources disponibles
- Utiliser jQuery :
  - Télécharger à www.jquery.com
  - ou utiliser un CDN (Content Delivery Network); voir par exemple *https://developers.google.com/speed/libraries/devguide?hl=fr*

```
<script
        src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js">
</script>
```

Sélectionner avec jQuery

- L'instruction $(selecteur), ou jquery(selecteur), retourne le ou les éléments sélectionnés.

```
<h1 id="test"></h1>
$("#test").text("premier essai");
```

- L'instruction $(document).ready() permet d'exécuter du code (en argument de ready) lorsque le document DOM est totalement chargé.

```
$(document).ready(function() {
        $("#test").text("premier essai");
});
```

Sélectionner avec jQuery

```
<body>
    <ul id="menu">
        <li class="item"><p>This is a paragraph</p></li>
        <li class="item">No paragraph here</li>
        <li class="item">No paragraph here</li>
    </ul>
    <p>Email: <input type="email" id="email" /></p>
    <p>Plain Text: <input type="text" id="text" /></p>
    <script src="//ajax.googleapis.com/.../jquery.min.js"></script>
    <script src="my.js"></script>
</body>

$(document).ready(function(){
    $(".item").css("color", "blue");
    $("#menu .item p").css("color", "red");
    $("input[type=email]").css("border", "10px solid blue");
});
```

Sélectionner des éléments avec jQuery

Sélecteur	Sens	Exemple
:eq()	élément à la position donnée	$("li:eq(2)")
:gt()	éléments aux positions supérieures	$("li:gt(2)")
:lt()	éléments aux positions inférieures	$("li:lt(2)")
:first	premier élément matché	$("tr:first")
:last	dernier élément matché	$("tr:last")
:even	éléments aux positions paires	$("tr:even")
:odd	éléments aux positions impaires	$("tr:odd")
:animated	éléments en cours d'animation	$("p:animated")
:selected	éléments sélectionnés	$("option:selected")
:visible	éléments visibles	$("p:visible")
:hidden	éléments cachés	$("p:hidden")
:has()	éléments contenant	$("div:has(p)")
:parent	éléments parents	$("p:parent")

Créer des éléments avec jQuery

- L'instruction $(element) crée un nouvel élément.

```
<div id="container"> </div>
        $(document).ready(function(){
        var time = new Date().getHours();
        var elem = $("<h1>").attr("id","greeting").hide();
        if (time < 12) elem.text("Good Morning");
        else elem.text("Good Afternoon");
        $("#container").append(elem);
        $("#greeting").show("slow");
});
```

```
Il est possible d'écrire :
        if (time < 12)
                elem = $("<h1 id='greeting'>Good Morning</h1>").hide();
```

Quelques méthodes

- Les méthodes suivantes sont couramment utilisées :
  - text() : get/set le contenu textuel de l'élément
  - html() : get/set le contenu html de l'élément
  - css() : get/set les propriétés CSS
  - attr() : get/set les attributs de l'élément
  - hide(), show() et toggle() : cache ou rend visible l'élément
  - fadeIn(), fadeOut() et fadeToggle() : pour jouer avec l'opacité
  - slideUp(), slideDown() et slideToggle() : pour un effet glissant
  - addClass(), removeClass(), hasClass() et toggleClass() permettent de modifier dynamiquement la valeur de l'attribut class

La plupart permettent d'effectuer du method chaining

Exemple avec attr()

```
<ul>
	<li id="contact"><a>Toto</a></li>
</ul>

$(document).ready(function(){
	$("#contact a").attr({
		"href" : "http://www.toto.com/",
		"title" : "Visit Toto website",
		"id" : "atoto"
	});
	var newItem = $("<li>").text($("#contact a").attr("title"));
	$("ul").append(newItem);
});
```

Exemple avec css()

```
<h1>Beginning HTML and CSS</h1>
<p>
	<strong>The border property is:</strong>
	<span id="result"></span>
</p>

$(document).ready(function(){
	$("h1").css({
		"font-size" : "200%",
		"color" : "#ffffff",
		"height" : "100px",
		"width" : "500px",
		"background-color" : "#61b7ff",
		"border" : "10px solid #003366"
	});
	$("#result").text($("h1").css("border"));
});
```

Exemple avec html()

```
$(document).ready(function(){
	var methods = [ "attr()", "css()", "html()", "addClass()",
		"removeClass()", "hasClass()", "toggleClass()" ];
	var list = "";
	for ( var i = 0, len = methods.length; i < len; i++ )
		list += "<li>" + methods[ i ] +"</li>";
	$("#methods").html(list);
});
```

Exemple avec toggleClass()

```
.selected {
	background : #666;
	color : #fff;
}
<ul>
	<li id="home" class="item">Home</li>
	<li id="about" class="item">About</li>
	<li id="contact" class="item">Contact</li>
</ul>

$(document).ready(function(){
	$("#home").on("click", function(){
		$("#home").toggleClass("selected");
	});
});
```

Exemple avec show() et toggle()

```
$(document).ready(function(){
	$("#slow").show("slow");
	$("#fast").show("fast");
	$("#ms").show(1500);
	$("#toggle").on("click", function(){
		$("#toggled").toggle();
	});
});
```

Exemple avec slideToggle()

```
<dl>
	<dt>Term </dt>
	<dd>
		<ul>
			<li>item 1</li>
			<li>item 2</li>
			<li>item 3</li>
		</ul>
	</dd>
	...
</dl>

$(document).ready(function(){
	$("dt").on("click", function(){
		$(this).next().slideToggle();
	})
});
```

Gestion des évènements

- Méthodes importantes :
  - on() pour enregistrer un écouteur
  - off() pour supprimer un écouteur
  - trigger() pour déclencher un évènement

  en remplacement depuis jQuery 1.7 de click(), submit(), live(), die(), delegate(), undelegate(), bind() and unbind()

```
$(document).ready(function(){
	function toggler(){
		$(this).next().slideToggle();
	}
	$("dt").on("click", toggler;)
	$("button").on("click", function(){
		$("dt").trigger("click").off("click", toggler);
	});
});
```

Méthode on() a trois arguments

- Le second paramètre désigne le type d'éléments qui est concerne par l'écouteur (même si un élément de ce type est créé plus tard).

```
$(document).on( "click", "a", function(){
      //code goes here
});
```

"Listen to every click on the whole document, and if it happens on an <a> element, re this event."

jQuery UI

- jQuery UI:
  - collection de GUI widgets
  - download a www.jqueryui.com ou utiliser un CDN

```
<head>
      <meta charset="utf-8">
      <title>jQuery UI Draggable and Droppable</title>
      <link rel="stylesheet"
      href="http://ajax.googleapis.com/ajax/libs/jqueryui/1.11.1/themes/
      smoothness/jquery-ui.css" />
</head>
<body>
...
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js">
</script>
<script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.11.1/jquery-ui.min.js">
</script>
</body>
```

Drag and Drop

```
<div id="container">
	<div id="draggable">
		<p>Drag here</p>
	</div>
	<div id="droppable">
		<p>Drop here</p>
	</div>
</div>

$(document).ready(function(){
	$("#draggable").draggable();
		$("#droppable").droppable( {
		drop: function(event, ui) {
			$(this).css("border","4px solid").html("<p>Dropped!</p>");
		}
	});
});
```

Resizable ans Selectable

```
<div id="resizable"> <p>Resizable Element</p> </div>

$(document).ready( function(){
	$("#resizable").resizable();
});

<ol id="selectable">
	<li>These items are selectable </li>
	<li>These items are selectable </li>
	<li>These items are selectable </li>
</ol>

$(document).ready(function(){
	$("#selectable").selectable();
});
```

Sortable

```
<ol id="sortable">
	<li>One</li>
	<li>Two</li>
	<li>Three</li>
	<li>Four</li>
	<li>Five</li>
</ol>

$(document).ready(function() {
	$("#sortable").sortable().disableSelection();
});
```

- Et tous les autres widgets sur http://jqueryui.com/. En particulier :
  - accordion
  - tabs