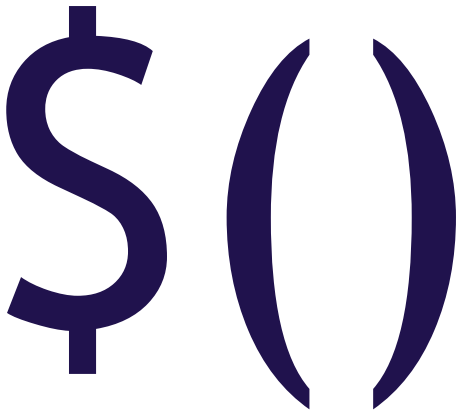
Cours/TP jQuery

CIFACOM - Julien Guézennec - (MAJ) 18/04/2013



Pour me présenter...

Je réalise des sites et applications Internet/multimédia depuis 15 ans (1998).

J'ai commencé par la post-production vidéo et 3D, comme intermittent du spectacle. Ensuite, après avoir travaillé 4 ans dans une agence Web, comme **développeur back/front office** LAMP/JS/HTML/SQL/FLASH... Je suis allé un an en Bolivie pour un projet de bibliobus puis j'ai vécu 1 an comme **freelance "concepteur/Developpeur web"**.
Depuis 2008 je travaille chez Bouygues Telecom, comme **Ingénieur concepteur/développeur Web** dans un laboratoire d'innovation R&D "convergence Internet" : Internet sur TV (Set Top Box / TV connectés), Internet mobile, tablettes tactile, etc... J'y réalise principalement des **prototypes de WebApps**

Depuis 2007 je développe aussi des quelques **plugins jQuery** et j'entretiens une base de connaissance
Depuis Janvier 2011, je fais activement parti d'une **communauté de développeurs** JavaScript : http://parisjs.org (C'est open !)

Mes langages et mots clefs : jQuery HTML5 CSS3 PHP MySQL Flex/Flash/Air LAMP SDK Android JavaScript Animation Logo Design HTML5 jQuery User-Experience UX XHTML ActionScript Web2.0 User-Interface Design Web-Development Web-Design CSS3 Mobile Devices Wordpress

Contacts : @molokoloco | julien-guezennec@cifacom.com | http://b2bweb.fr

---

Sommaire

COURS

- Les outils de développement
- jQuery ???
- jQuery -in practice-
- Documentation jQuery
- Cibler et modifier des éléments
- -Parenthèse- "this" is out of control!
- Evènements utilisateur et dynamiques
- Animations ("Effects")
- Les Formulaires
- Les requêtes "AJAX"
- "Promise" et "Deferred" ?
- Les bases d'un plugin jQuery
- Plugins jQuery utiles

EXERCICES / TP

- Navigation par onglet
- Validation de formulaires
- Concept de carrousel

Les outils de développement

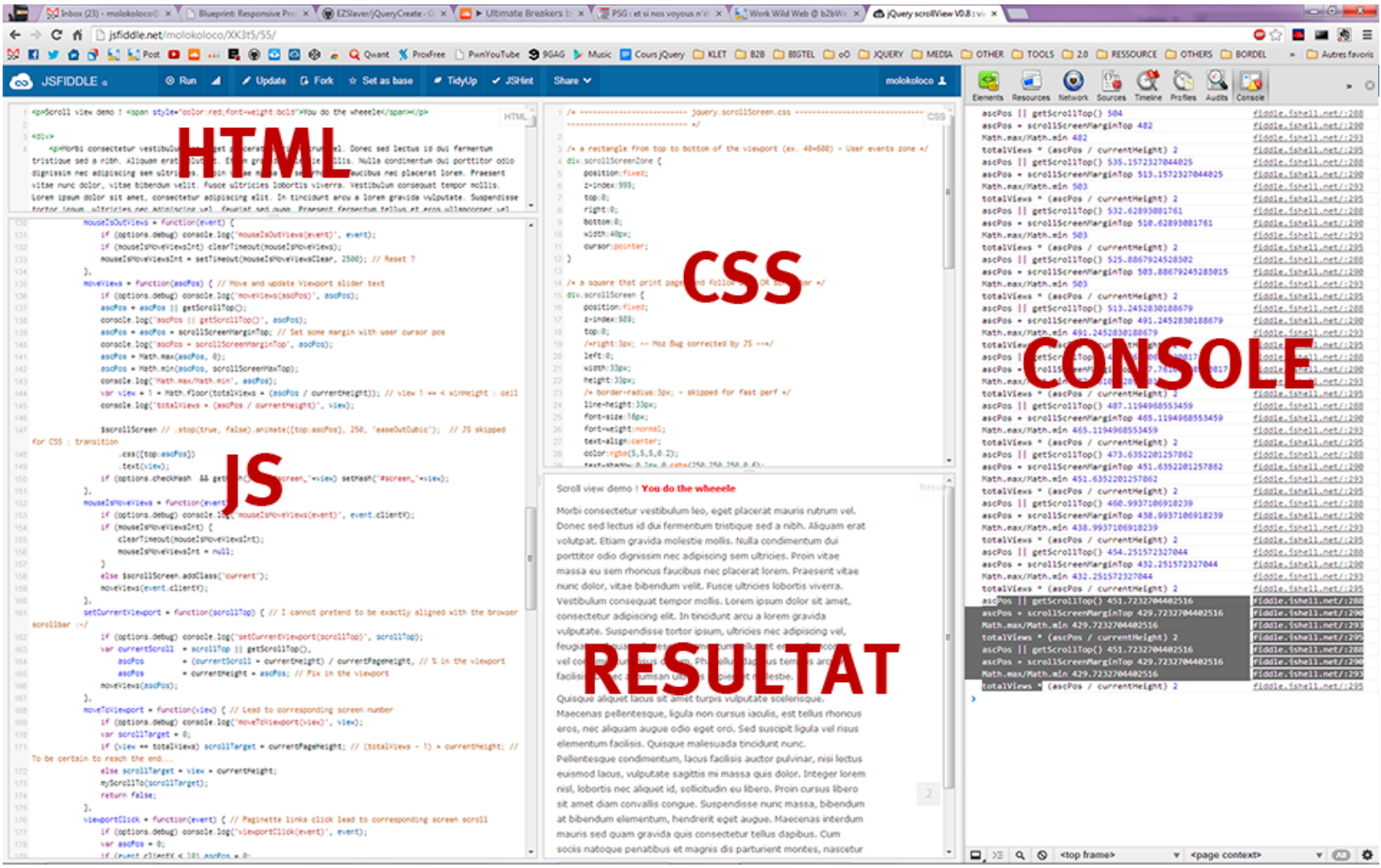
Offline :

- http://www.adobe.com/products/**dreamweaver**/
- http://www.aptana.com/
- http://www.jetbrains.com/webstorm/
- http://wiki.netbeans.org/JavaScript
- http://www.**eclipse**.org/
- http://www.**sublimetext**.com/
    - Installer et tuner Sublime Text aux p'tits oignons http://bit.ly/st-webfront
- http://emea.microsoftstore.com/fr/fr-FR/Microsoft/Creation-+-Developpement/Visual-Studio-2012
- ...

Online :

- http://**cloud9ide**.com
- http://**jsfiddle**.net
- http://codepen.io/pen/
- http://jsdo.it
- http://jsbin.com
- http://cssdeck.com
- http://dabblet.com
- http://fiddlesalad.com/javascript/
- http://www.akshell.com/ide/
- ...

TIP : Pour ajouter jQuery dans n'importe quelle page :
http://www.learningjquery.com/2009/04/better-stronger-safer-**jquerify-bookmarklet**

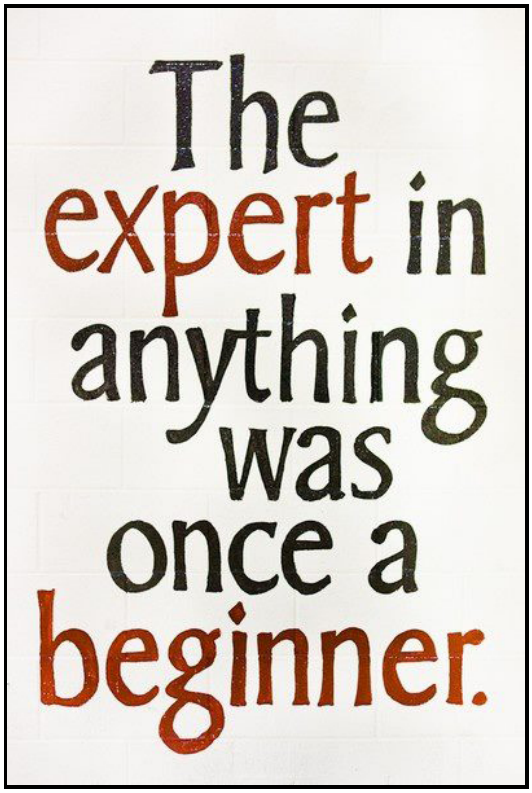


~(Parenthèse)~

La mémoire et le cloud : La connaissance est un flux en streaming...
…….."The community has gathered some good resources for you :D"…….

Au fil d'un parcours professionnel, on est enmené à accumuler une grande quantité de fichiers et ressources... pas de secret : Mieux vaut avoir de l'ordre !

Sur chaque projet il faut capitaliser, centraliser ces sources, sauvegarder
... et toujours améliorer la formule !
Ajouter des favoris sur Github, créer et entretenir son propre Wiki, Bookmarker tout, etc...



---

jQuery ???

➡ **jQuery** est une **bibliothèque JavaScript libre** qui porte sur l'interaction entre JavaScript (comprenant Ajax) et HTML, **et a pour but de simplifier des commandes communes de JavaScript**. La première version date de janvier **2006**.

jQuery est "DOM centrique" et a principalement été créé dans le but de :

- **manipuler des pages (DOM + CSS) dynamiquement**
- **prendre en compte des données externes (AJAX)**

> *"jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript."*

jQuery 1.8 se présente comme un **unique fichier JavaScript** de **91.2KB** en version minimifié et **32.64KB** (min + gzip) contenant toutes les fonctions de base. Il peut être inclus dans toute page web en utilisant le code suivant :

```
<script src="http://code.jquery.com/jquery-1.9.1.min.js" type="text/javascript"></script>
```

jQuery est disponible en plusieurs versions :

- http://blog.jquery.com/2013/01/15/jquery-1-9-final-**jquery-2**-0-beta-migrate-final-released/

➡ Ses fonctions principales

- **Sélecteurs** : Parcourir le DOM avec les sélecteurs CSS 1 à 3 et un support basique de XPath. Filtrer les résultats
- **Elements et attributs** : Changer un élements ou avoir des informations sur lui
- **Evènements :** Actions utilisateurs ou de modules JS
- **CSS :** Manipuler les styles
- **Animations / Effets :** Changer l'état d'un élement dans l'instant ou sur la durée
- **AJAX :** Requêtes externes à la page
- **Plugins :** Extensions spécifiques au cas par cas
- **Fonctions pratiques :** Détection navigateur, manipulation d'objets JS, *deferred*, etc

Exemples comparatifs entre le JavaScript natif et la bibliothèque jQuery

```
var  element = document.getElementById('elementId');           // Natif
var $element = $('#elementId');                                // jQuery

var  elements = document.getElementsClassName('elementClass'); // Natif
var $elements = $('.elementClass');                            // jQuery
```

Requête Ajax en JavaScript “natif” :

```
function load(url, callback) {
        var xhr;
        if(typeof XMLHttpRequest !== 'undefined') xhr = new XMLHttpRequest();
        else {
            var versions = ["MSXML2.XmlHttp.5.0",
                            "MSXML2.XmlHttp.4.0",
                            "MSXML2.XmlHttp.3.0",
                            "MSXML2.XmlHttp.2.0",
                            "Microsoft.XmlHttp"]
             for(var i = 0, len = versions.length; i < len; i++) {
                try {
                    xhr = new ActiveXObject(versions[i]);
                    break;
                }
                catch(e){}
             }
        }
        xhr.onreadystatechange = ensureReadiness;
        function ensureReadiness() {
            if(xhr.readyState < 4) return;
            if(xhr.status !== 200) return;
            // all is well
            if(xhr.readyState === 4) callback(xhr);
        }
        xhr.open('GET', url, true);
        xhr.send('');
}
// Our simplified "load" function accepts a URL and CALLBACK parameter.
load('http://www.monsite.com/serveur.php', function(xhr) {
```

```
    document.getElementById('container').innerHTML = xhr.responseText;
});
```

Requêtes Ajax avec jQuery :

```
$.ajax({
    url: 'http://www.monsite.com/serveur.php',
    success: function(data) {
        $('#container').html(data);
    },
    error: function(jqXHR, textStatus, errorThrown) {
        console.log(textStatus);
    }
});

// ou encore...

$.get('http://www.monsite.com/serveur.php').done(function() {
     alert('OK');
});
```

Autres Frameworks de la famille jQuery :

JQUERY UI

- http://jqueryui.com/
- http://jqueryui.com/demos/
- Alternative : http://jquerytools.org/demos/

JQUERY MOBILE

- http://jquerymobile.com/
- http://jquerymobile.com/demos/1.1.1/
- http://tuts.pinehead.tv/2012/04/16/jquery-mobile-development-guide/

Frameworks alternatifs à jQuery :

La concurrence directe...

- ~~http://prototypejs.org~~
- ~~http://script.aculo.us~~
- **http://zeptojs.com**
- http://microjs.com
- **http://dojotoolkit.org**
- **http://mootools.net**
- http://yuilibrary.com
- ...

Outils/Frameworks plus puissants (Orientés software et MVC)

- **http://www.sencha.com/products/extjs**
- http://www.wakanda.org
- http://factory.joshfire.com
- …

MVC, data binding ???

- http://sporto.github.io/blog/2013/04/12/comparison-angular-backbone-can-ember/
- http://codebrief.com/2012/01/the-top-10-javascript-mvc-frameworks-reviewed/
    - http://emberjs.com
    - http://backbonejs.org
    - http://angularjs.org
    - http://canjs.com
    - http://knockoutjs.com
    - http://sproutcore.com

Cas à part...

- http://nodejs.org (ServerSide JS)
- http://coffeescript.org (Neo JS)
- https://github.com/mrdoob/three.js/ (3D & WebGL)
- https://github.com/ariya/phantomjs (Web Scrapping)
- http://twitter.github.com/bootstrap/ (Responsive UI)
- …

jQuery -in practice-

➡ Un premier exemple de fichier utilisant jQuery

Code à mettre dans un fichier en local sur votre ordinateur puis à éxécuter en "localhost" dans le navigateur...
(Ressources : cf. http://jquery.com/download/ ou https://developers.google.com/speed/libraries/devguide)

Source de départ :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="utf-8">
    <title>TP jQuery : Demo</title>
</head>
<body>
```

```
    <h1>Démo</h1>
    <p><a href="http://jquery.com/">jQuery</a></p>

    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js"
type="text/javascript"></script>

    <script type="text/javascript">
    $(document).ready(function(){
        $("a").click(function(event){
            event.preventDefault();
            alert("As you can see, the link no longer took you to jquery.com");
        });
    });
    </script>

</body>
</html>
```

Un exemple complet (qui fait peur et) qui montre jQuery en action

- http://jsfiddle.net/molokoloco/8t6Qk/

```
$('<img/>')                  // Crée un fragment HTML pour une balise image
    .attr('src', 'http://mozorg.cdn.mozilla.net/media/img/home/firefox.png') // Attribut src de
la balise IMG
    .error(function(e) {     // Callback de gestion d'erreur
        alert('Erreur !', e);
    })
    .load(function(event) {  // Callback éxécuté lorsque l'image est chargée
        $(this)              // Référence à l'élément actuel qui reçoit le callback (<img>)
            .css({           // Change les attributs CSS de départ
                position : 'absolute',
                left     : '50%',
                top      : (($(window).scrollTop() || 0) - $(this).height())+'px', // Vertical
center
                margin   : '0 0 0 -' + ($(this).width() / 2) + 'px', // Horizontal center
                opacity  : 0
            })
            .animate({       // Crée une animation sur des propriétés CSS
                opacity  : 1,
                top      : ($(window).scrollTop() || 0) + 'px'
            }, 800)
            .delay(3000)     // Attend 3000 sec après la fin de l'animation
            .fadeOut(600, function () { // Animation de disparition
                $(this).remove();
            });
    })
    .appendTo('body');       // Ajoute au body (et donc commence le chargement de l'image)
```

Inclure jQuery

```
<!-- Grab Google CDN's jQuery. fall back to local if necessary -->
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
<script> !window.jQuery && document.write('<script
src="./js/jquery-1.7.1.min.js"><\/script>'); </script>
```

Il existe de nombreux CDN proposant jQuery (jQuery, Microsoft, Google, etc...)
Le site officiel pour télécharger : http://jquery.com/download/

➡ Initialiser jQuery au démarrage d'un document

```
// When jQuery is ready (document DOM is ready)
$(function(){
    // le code principale pour la page
});

// ...ou bien, identique
$(document).ready(function(){
    $('a').click(function(){ alert('click'); });
});

// When all is loaded, including external files (images...)
$(window).on('load', function()  {
    /* ... */
});
```

JQuery au complet !

http://api.jquery.com

- Ajax : The jQuery library has **a full suite of AJAX capabilities**. The functions and methods there in allow us to load data from the server without a browser page refresh
- Attributes : These methods **get and set DOM attributes of elements.**
- Callbacks Object : **manage callback** lists. It supports adding, removing, firing, and disabling callbacks
- Core
- CSS : These methods **get and set CSS-related properties** of elements
- Data : These methods allow us to **associate arbitrary data with specific DOM elements**
- Deferred Object : The **Deferred object**, introduced in jQuery 1.5, is **a chainable utility object**
- Dimensions : These methods are used to **get and set the CSS dimensions** for the various properties
- Effects : The jQuery library provides several techniques for **adding animation** to a web page. These include simple, standard animations that are frequently used, and the **ability to craft sophisticated custom effects**.
- Events : These methods are used to **register behaviors to take effect when the user interacts** with the browser, and to further manipulate those registered behaviors.
- Forms : These methods and **event handlers handle forms** and their various elements.
- Internals
- Manipulation : All of the methods in this section **manipulate the DOM** in some manner. A few of them simply **change one of the attributes of an element** (also listed in the Attributes category), while others set an element's style properties (also listed in the CSS category). Still others modify entire elements (or groups of elements) themselves—**inserting, copying, removing, and so on**. All of these methods are referred to as "setters," as they change the values of properties.
  A few of these methods—such as `.attr()`, `.html()`, and `.val()`—also act as "getters," **retrieving information from DOM elements** for later use.

- Miscellaneous
- Offset : **Get the current coordinates** of the first element
- Properties : Some **properties**
- Selectors : Borrowing from CSS 1–3, and then adding its own, **jQuery offers a powerful set of tools for matching a set of elements** in a document.
  To use any of the meta-characters (such as `!"#$%&'()*+,./:;<=>?@[\]^`{|}~` ) as a literal part of a name, it must be escaped with with two backslashes: `\\`. For example, an element with `id="foo.bar"`, can use the selector `$("#foo\\.bar")`. The W3C CSS specification contains the complete set of rules regarding valid CSS selectors. Also useful is the blog entry by Mathias Bynens on CSS character escape sequences for identifiers.
- Traversing : DOM an **jQuery elements collections traversal**
- Utilities : Tools and **usefull methods** or properties

➡ Quel est la logique d'utilisation de jQuery ?

6 exemples représentatifs de l'usage :

```
// Exemple n°1 :
// Mise en cache d'une collection d'éléments jQuery pour réutilisation
var $e = $(element); // var $menuLinks = $('ul#menu a');

// Exemple n°2 :
// Appel d'une méthode sur un élément jQuery pour obtenir une valeur
var x = $(element).methode(); // var menuWidth = $('ul#menu').width();

// Exemple n°3 :
// Les différents type d'arguments lors de l'appel d'une méthode
$(e).methode(value1);                       // $('a.menu').width('100px');
$(e).methode(prop1, value1);                // $menu.attr('href', myUrl);
$(e).methode({prop1:value1, prop2:value2});// $menu.attr({href:myUrl, alt:''});
$(e).methode(obj, function() {});           // $menu.animate(props, callback);
$(e).methode1().methode2().methode3();      // $menu.fadeOut(0).width(10).fadeIn();

// Exemple n°4 :
// "lenght" est l'unique propriété des collections d'éléments jQuery
var totalElements = $(e).lenght;

// Exemple n°5 :
// Certaines "propriétés" utiles sont aussi disponibles avec jQuery
var browserInfos = $.browser;
var jQueryVersion = $.fn.jquery;

// Exemple n°6 :
// Certaines méthodes jQuery ne s'utilisent pas autour de collections d'éléments
var req = $.ajax({url: domainUrl});
var getUrlEncoded = $.param( obj );
$.noConflict();

$.fn.myPlugin = function() { return this; }
// $(e).myPlugin()
```

Exemple de gestion d'un composant d'interface avec onglets :

- http://jsfiddle.net/molokoloco/e3Lhc/
  (Cf. http://learn.shayhowe.com/advanced-html-css/jquery)

➡ Les arguments de jQuery() ou... $()

Les astuces de jQuery et jQuery "core" : http://api.jquery.com/jQuery/
Dans l'arrière cuisine... https://github.com/jquery/jquery/blob/master/src/core.js

- **Accepts a string containing a CSS selector** which is then used to **match a set of elements.**
  - $(**selector** [, context])
    ```
    $('#monId')
    $('.maClass', $parent)
    ```
  - $(**element**)
    ```
    $(document.getElementById('test'))
    ```
  - $(**object**)
    ```
    $(window)
    ```
  - $(**elementArray**)
    ```
    $(document.getElementsByClassName('test'))
    ```
  - $(**jQuery object**)
    ```
    $($element)
    ```
- **Creates DOM elements** on the fly from the provided string of raw HTML.
  - $(**html** [, ownerDocument])
    ```
    $('<p>Texte</p>')
    $('<p/>')
    $('<p>Texte <b>Yep</b></p>')
    ```
  - $(**html, props**)
    ```
    $('<p/>', {text:'Texte', 'class':'red'})
    ```
- Binds a function to be executed **when the DOM has finished loading.**
  - $(**callback**)
    ```
    $(function() { /* DOM + jQuery Ready */ })
    ```

Intégrer jQuery dans un projet web ?

Par chance, jQuery est le plus populaire des Frameworks JavaScript. De très nombreux projets sont donc prévus pour fonctionner parfaitement avec lui.

Exemple d'intégration :

- **http://html5boilerplate.com**
  *"HTML5 Boilerplate" : The web's most popular **front-end template**.*
  HTML5 Boilerplate helps you build fast, robust, and adaptable web apps or sites. Kick-start your project with the combined knowledge and effort of 100s of developers, all in one little package.

- ➡ **http://twitter.github.com/bootstrap/**
  Sleek, intuitive, and powerful front-end framework for faster and easier web development.
  By nerds, for nerds : Built at Twitter by @mdo and @fat, Bootstrap utilizes LESS CSS, is compiled via Node, and is managed through GitHub to help nerds do awesome stuff on the web.
  Made for everyone : Bootstrap was made to not only look and behave great in the latest desktop browsers (as well as IE7!), but in tablet and smartphone browsers via responsive CSS as well.
  Packed with features : A 12-column responsive grid, dozens of components, JavaScript plugins, typography, form controls, and even a web-based Customizer to make Bootstrap your own.

Si par malheur un Framework éxistant du projet utilise déjà la variable "**$**" (Risque de conflit) :
>>> http://api.jquery.com/jQuery.noConflict/

Exemples variés avec jQuery

- http://jsfiddle.net/molokoloco/4yGUw/ (Square logos with google font)
- http://jsfiddle.net/molokoloco/uQpQF/ (jQuery $.map())
- http://jsfiddle.net/molokoloco/9yksd/ (Add an element in a list)

Documentation jQuery

Le site officiel ! http://jquery.com
Le "coeur" de jQuery expliqué ici : http://api.jquery.com/jQuery/

➡ La documentation officielle

Documentation (Passer 2 jours à lire la doc ??? ;)

- http://learn.jquery.com/about-jquery/**how-jquery-works/**
- ➡ **http://api.jquery.com**
- **http://learn.jquery.com/**
- **http://docs.jquery.com/Main_Page**
- **http://api.jquery.com/visual/**

Les documentations non-officielles

- http://remysharp.com/jquery-api/
- http://jsapi.info/
- http://jqapi.com/
- ...

Les autres ressources du site :

- Tutoriaux : http://docs.jquery.com/Tutorials
- Bugs Traker : http://bugs.jquery.com/newticket
- Discussions : http://docs.jquery.com/Discussion
- Forums : http://forum.jquery.com/

Le code source de jQuery

- **https://github.com/jquery**

- https://github.com/jquery/jquery

jQuery 1.7 Cheat Sheet

- **http://woorkup.com/wp-content/uploads/2011/12/jQuery-17-Visual-Cheat-Sheet1.pdf**

Les développeurs derrière jQuery...

- http://ejohn.org (Father of jQuery)
- http://addyosmani.com
- http://brandonaaron.net
- http://benalman.com
- ...

Autres ressources

D'abord, lorsque l'on cherche **sur Google, préfixer la requête avec "jQuery " + mot-clef**
>> C'est mieux de connaitre le nom exact (technique) du composant !

- carousel, tab, date picker, infinite-scroll, etc...

… évident mais j'insiste ;)

Par exemple, rechercher "jquery carousel ajax" et "jquery carousel responsive" et "jquery carousel 3D" puis s'amuser à combiner les trois meilleurs plugins !!!
:)

CONSOLE et DEBUGGING !

- Cf. cours JavaScript https://docs.google.com/document/d/1j9KsH-YtlYuMhmcPRlqtNJ_JdrD2JUiErmwEUTWt23I/edit?usp=sharing
- **http://fixingthesejquery.com** !

A FAIRE !!!

Des exercices intéractifs en lignes :

- http://jqfundamentals.com/chapter/jquery-basics
- Plus une collection des meilleurs ressources, **en bas de ce document**...

Une note spéciale au workshop d'introduction à jQuery (par son créateur) :

- http://ejohn.org/apps/workshop/intro/

Un point sur la Syntaxe !

Le JS de base... (Cf. cours précédant !!!)
https://docs.google.com/document/d/1j9KsH-YtlYuMhmcPRlqtNJ_JdrD2JUiErmwEUTWt23I/edit?usp=sharing

Cibler et modifier des éléments

Via une syntaxe claire, mais qui peut porter à confusion pour les débutants, nous allons **manipuler des ensembles d'éléments** dans un exercice pratique...

*En JS, on est dans l'**arbre DOM**,*
bien plus qu'un simple "string" HTML
(que peut voir PHP par exemple...)

➡ Les bases !!!

Les éléments sélectionnés via jQuery sont retourné sous la forme d'un tableau (collection d'éléments)

```
$(window) || $(document)  // Objets (natifs)... (obj)
$('body') || $('form')    // Tags HTML ("tag")
$('*')                    // Tous les tags HTML (de la page)("*")
$('#monId')               // Element(s) avec identifiant "ID" ("#")
$('.maClass')             // Element(s) avec classe CSS
$('button.maClass')       // Element(s) avec tag et classe CSS (".")
$('input[type=password]') // Seulement avec un certain attribut ("tag[attr=val]")
$('body:first-child')     // Pseudo sélecteur... (":")
$('a:first')              // Pseudo sélecteur... (":")
$('div:not(:animated)')   // Deux pseudos sélecteurs avec négation ("not()")
$('p a')                  // Enfant(s) : lien(s) "a" dans des "p"(" ")
$('.maClass1, .maClass2') // Plusieurs éléments avec propriétés différentes (",")
```

GET et EQ

jQuery ajoute de nombreuses méthodes autour de ces éléments natif. On dit que jQuery "*proxifie*" les éléments (natifs du DOM) : les méthodes natives ne sont plus accessibles directement.

```
// WRONG
$("#foo").innerHTML = "I clicked it";

// RIGHT (If necessary)
$("#foo").get(0).innerHTML = "I clicked it";

// SIMPLY RIGHT !
$("#foo").html("I clicked it");
```

Certaines méthodes jQuery s'appliquent à un ensemble d'éléments et d'autres sur un seul élément
Il peut être pratique de choisir un élément en particulier dans une collection

- **eq() :** http://api.jquery.com/eq/
  Reduce the set of matched elements to the one at the specified index

```
var $secondLi = $('ul > li').eq(1);
$secondLi.html('Ok'); // "$secondLi" is the second LI in the matched set
```

- **get() :** http://api.jquery.com/get/
  Retrieve the DOM elements matched by the jQuery object.

```
var domSecondLi = $('ul > li').get(1);
domLi2.innerHTML = 'Ok'; // "domLi2" is the native second LI element
```

Exemples :

```
// <p class="maClass"></p>
$('p.maClass').html('T<em>es</em>t');
```

```
// <p class="maClass">T<em>es</em>t</p>

$(this).parent().html('T<em>es</em>t');

// <p><strong>This</strong> <strong>is a test</strong></p>
$('strong:first').addClass('bigText');
// <p><strong class="bigText">This</strong> <strong>is a test</strong></p>

// Obtenir une valeur
var imgAttrWidth = $('img#logo').attr('width');
// 100%
var imgWidth = $('img#logo').width();
// 923 (px)

// Change un attribut du body
$('body').attr('height', '100%');
// <body height="100%">

// Change les attributs height et width d'une image
$('img#logo').attr({height: 120, width: '100%'});
// <img id="logo" height="120" width="100%" src="logo.png"/>

// Change le style height et width d'une image
$('img#logo').css({height: 120, width: '100%'});
// <img id="logo" src="logo.png" style="height:120px;width:100%;"/>

// Ecouter une évenement sur un element
$('img').on('load', function() { alert("load"); });

// Vide le contenu d'une balise
$('h1').empty();
$('h1').html('');

// Efface une balise
$('h1').remove();
$('h1').detach();

// Insert un élément à la fin d'une liste
$('ul li:last').after('<li>Added text ' + p + '</li>');

// Vérifer qu'un élément posséde un critère (Id, Classe, Pseudo sélecteur, etc...)
if ($(this).is(':first-child')) {}
if ($(this).is('a')) {}

// Si le lien est visible ?
if ($('#link').is(':visible') == 'true') {}

// Obtenir la valeur d'un champs de formulaire (input, select, textarea, etc...)
$('#myselect').val();

// Syntaxe spécifique pour le select, idem que : $('#myselect').val();
$('#myselect option:selected').text();

// Si le lien existe ?
// (la collection jQuery contient plus que 0 élément ? : 0 == false)
```

```
if ($('#link').length) {}

// Affiche les URL des liens qui sont dans les paragraphes
$('p a').each(function(index, element) { // element == $('p a')[index]
    $(element).after(' ['+$(element).attr('href')+'] ');
});

// Parcourir un tableau de valeurs
$.each(['tutut', 'totot'], function(index, valeur) {
    console.log(index, valeur);
});

// Gérer le focus avec des valeurs par défaut sur un champs "input" (idem attribut
"placeholder" HTML5)
var $myInput = $('input#search');
$myInput
    .data('valInit', $myInput.val())
    .focus(function() {
        if ($myInput.val() == $myInput.data('valInit'))
            $myInput.val('');
    })
    .blur(function() {
        if ($myInput.val() == '')
            $myInput.val($myInput.data('valInit'));
    });

// Masquer les div non animées
$('div:not(:animated)').fadeOut();

// Masquer les div qui n'ont pas la classe "active"
$('div:not(.active)').fadeOut();

// Créer une div de "popup" dynamique
var popUp = function(message, delay) {
       delay = delay || 5000;
       $('<div />')
           .addClass('ui-corner-all')
           .css({opacity: 0.96, top: $(window).scrollTop() + 100})
           .html('<h1>Message...</h1><p>'+message+'</p>')
           .appendTo('body')
           .hide()
           .fadeIn(200)
           .delay(delay)
           .fadeOut(400, function(){ $(this).remove(); });
};

// Update d'un tag image
$('img.hate').attr({
    src: '/images/love.gif',
    alt: 'jQuery Logo'
});

// Click outside div ?
$('body').click(function(event) {
    if (!$(event.target).is('div') && !$(event.target).parents().is('div')) {
         $(this).fadeOut().fadeIn();
```

```
    }
});

// Working on a collection
$('p').css('color', function(index) {
    return 'hsl(' + (Math.random() * 360) + ', 100%, 50%)';
});
```

A noter...

New elements that match a selector **do not automagically appear in existing collections**

```
var $li = $("#wheatBeers li");
// $li.length == 3;

$("<li>", {text:"Arh Zo !"}).appendTo("#wheatBeers");
// guess what? $li.length == 3
```

Every call to jQuery's traversal/selection methods returns a new jQuery object
That's why **you can't compare jQuery objects** for equality

```
$("#foo") === $("#foo")       // false
$("#foo")[0] === $("#foo")[0] // true
$("#foo").is("#foo")          // true
```

Exemples de propriétés utilisables avec “$(e).css(props)”

```
{
        "position": "absolute",
        "display": "block",
        "float": "",
        "zIndex": 500,
        "overflow": "hidden",
        "whiteSpace": "",
        "textOverflow": "",
        "cursor": "pointer",
        "textAlign": "center",
        "maskImage": "",
        "backgroundImage": "",
        "top": 0,
        "left": 0,
        "marginTop": "",
        "marginLeft": "",
        "width": 160,
        "height": 30,
        "transform": "",
        "transformOrigine": "",
        "transformStyle": "preserve-3d",
        "backfaceVisibility": "visible",
        "minWidth": "",
        "maxWidth": "",
        "minHeight": "",
        "maxHeight": "",
        "right": "",
```

```
        "bottom": "",
        "marginRight": "",
        "marginBottom": "",
        "padding": "",
        "backgroundPosition": "",
        "backgroundSize": "100% 100%",
        "backgroundRepeat": "no-repeat",
        "backgroundAttachment": "",
        "backgroundOrigin": "border-box",
        "backgroundClip": "border-box",
        "backgroundColor": "rgba(255, 255, 0, 1)",
        "clip": "",
        "boxSizing": "",
        "boxShadow": "1px 1px 3px rgba(30,30,0,0.8)",
        "boxReflect": "",
        "border": "1px solid white",
        "borderImage": "",
        "borderWidth": "",
        "borderRadius": "20px / 50px",
        "outline": "",
        "outlineOffset": "",
        "color": "navy",
        "textShadow": "",
        "textStroke": "",
        "fontSize": "25px",
        "letterSpacing": "",
        "wordSpacing": "",
        "textIndent": "",
        "lineHeight": "30px",
        "opacity": 1,
        "maxSize": ""
}
```

➡ Les “GET” (“getters”) et les “SET” (“setters”)… ?

Certaines **méthodes jQuery permettent soit de récupérer la valeur (GET) d’une propriété soit de la définir (SET) :**

>>> **attr(), html(), val(), width(), prop()…**

```
// Getter... (Call it without argument)
var myWidth = $('#myselect').width(); // myWidth  == 10 (px)

// Setter (Call it with an argument)
$('#myselect').width('50%');

// Getter...
var myId = $('#myselect').attr('id'); // toto

// Setter
$('#myselect').attr('id', 'tutu');
```

Si on demande a jQuery (GET) la propriété *width()* alors que le sélecteur pointe sur une collection d’éléments, jQuery renvois uniquement la valeur du premier.

Si on fait un “SET” alors toute la collection recoit la nouvelle valeur (jQuery effectue l’équivalent d’une boucle “each()” sur la collection en interne...)

Pourquoi les “callback” ? `“a function as a handler”`

Il faut **encapsuler les instructions à effectuer plus tard** (Asynchrones) dans **une nouvelle fonction dédié** (anonyme ou non). Cela permet de ne pas effectuer les instructions tout de suite.

```
// WRONG
$("#foo").click(alert("I clicked it"));

// You just executed 'alert', and failed to pass a function as a handler. Huh?
// It's the same as doing the following, and just as broken.

var log = alert("I clicked it"); // log === undefined
$("#foo").click(log);

// RIGHT
var callAlert = function(e) {
    alert("I clicked it");
};
$("#foo").click(callAlert);

// RIGHT
$("#foo").click(function(e) {
    alert("I clicked it");
});
```

Exercice...

- http://jsfiddle.net/molokoloco/23RAr/

Cherchons sur la documentation...

Comment sélectionner le 2ème paragraphe d'une collection ?

$('.box > p ...')...css('background', 'rgba(0,0,0,0.2)');

Une ligne sur deux ?

$('.box > p ...')...

En relation, rapide survole...

- Catch/Find an element with CSS "xPath"
- SHORTCUT MEMO
- JQUERY SELECTORS (CSS3 like xPath)
- CUSTOM SELECTORS

- INSERT ELEMENT
- Elements methodes

-Parenthèse-

“this” is out of control!

$(e).f1().f2();

jQuery fonctionne par **chainage de méthodes**, chacunes renvoyant “this” (son **contexte / scope**), c’est à dire l’ensemble de l’objet jQuery de départ : $(e)

➡ “this” est très pratique :

```
$('a').click(function() {
    $(this).fadeOut();
});
```

mais il y a quelques surprises...

- **Event handlers, jQuery.fn.each, jQuery UI**
  Explicitly set this to be a DOM element
- **AJAX callbacks**
  Explicitly set this to be the options object
- **Random Plugins With Callbacks**
  Might set this, might not. Be on the lookout.
- **jQuery.each**
  Explicitly sets this to be the value being iterated, but weird.

```
$.each(["foo", "bar"], function(i, value) {
    // this == ["f","o","o"]
    // value == "foo"
});
```

jQuery.each on an array : Don't use this here, use the second argument

Just being aware of this fact is half the battle

You pass a lot of functions to various jQuery methods Event Handlers, AJAX callbacks, Utility methods, Plugin callbacks.
jQuery and plugins will change the 'this' (the scope) of these functions

```
$("#foo").click(function(e) {
    $.getJSON("/ajax/page/", function(data) {
        $(this).text(data.status); // IT'S NOT WORK
    });
});
```

This, in this example, gives new jQuery users fits.
Be very aware of this when writing object-oriented code
Don't drop the scope ! If this isn't the this you think this is, you can use the debugger and/or the console to confirm your suspicion.
Hmm, this doesn't look like a DOM element from here!

Let's get this straight...

Use a variable to alias this :

```
$("#foo").click(function(e) {
    var self = this;
    $.getJSON("/ajax/page/", function(data) {
        $(self).text(data.status); // ITS WORK NOW!
    });
});
```

jQuery.proxy(fct, scope)
Returns a reference to a function that will run with a particular context

```
var person = {
    name : "Ace S. McCloud",
    init : function() {
        $("#sayHi").click($.proxy(function(e) {
            // Now 'this' refers to the person object!
            // We can still use e.target to reference the element
            $(e.target).text(this.name);
        }, this));
    }
};
```

Une petite relecture du cours sur le “this” natif JS ?
“***this* keyword is one of the most confusing in JavaScript**”

- https://docs.google.com/document/d/1j9KsH-YtlYuMhmcPRlqtNJ_JdrD2JUiErmwEUTWt23I/edit?usp=sharing

Evènements utilisateur et dynamiques

➡ Les "Events" jQuery...

- http://api.jquery.com/category/events/

The standard events in the Document Object Model are :
blur, focus, load, resize, scroll, unload, beforeunload, click, dblclick, mousedown, mouseup, mousemove, mouseover, mouseout, mouseenter, mouseleave, change, select, submit, keydown, keypress, and keyup.

- **Browser Events**
- **Document Loading**
- **Form Events**
- **Keyboard Events**
- **Mouse Events**

- **Event Handler Attachment**
- **Event Object**

Fonctions de base pour la gestion des évènements :

- **on()** : Attach an event handler function for one or more events to the selected elements.
- **off() :** Detach an event handler
- **trigger() :** Execute an event handler
- Deprecated : bind() / live() / delegate() ...

Principe premier : Sur une collection d'éléments, on écoute un ou plusieurs évènements, puis on appelle une function (callback) lorsqu'il s'éxécute (trigger). Cf. http://api.jquery.com/category/events/

- $element.**on**(events, handler(eventObject))
- $element.**on**(events [, selector] [, data], handler(eventObject))

jQuery's event model **can trigger an event by any name on an element**, and **it is propagated up the DOM tree** to which that element belongs (See “Event bubbling” : https://docs.google.com/document/d/1j9KsH-YtlYuMhmcPRlqtNJ_JdrD2JUiErmwEUTWt23I/edit#)

il faut faire attention à la fréquence de chaque évènement.
Par exemple l’event “submit” risque d’être unique alors que l’évenement “mousemove” s'exécute à une fréquence très rapide :

> *“The mousemove event fires when the user moves the mouse pointer by 1 pixel or more in any direction in the browser window”*

On ne peut pas effectuer beaucoup d’opération dans la fonction de callback sur l’événement “mousemove” : le moteur JS peut être incapable d’effectuer trop de calcul à interval si rapproché. Par contre sur le “submit” il est possible de vérifier tous les inputs d’un formulaire...

Exemples :

```
$('#foo').on('click', function() {
    alert('User clicked on "foo."');
});

// Trigger an artificial click event
$('#foo').trigger('click');

$('#foo').on('mouseenter mouseleave', function(event) {
    console.log(event.type);
});

$('#foo').on({
    click: function() {
        // do something on click
    },
    mouseenter: function() {
        // do something on mouseenter
    }
});

$('#foo')
    .on('click' function() {
        // do something on click
    })
    .on('mouseenter', function() {
        // do something on mouseenter
    });

var divClicked1 = function(event) {
    // Celui qui ecoute un evenement (la DIV)
    console.log('event.currentTarget', event.currentTarget);
    // Celui qui reçoit (la DIV ou un de ces enfants : P, SPAN, ...)
    console.log('event.target', event.target);

    $(event.currentTarget).css({background: 'yellow'});
    $('div').off('click', divClicked1); // Delete the listener
};

$('#foo').on('click', divClicked1);
$('#foo').on('click', divClicked2);


// ADVANCED EXAMPLES...

// Trigger an artificial Keydown
var enterKeyEvent = $.Event('keydown', {
    which:   $.ui.keyCode.ENTER,
    keyCode: $.ui.keyCode.ENTER
});
$('input#search').trigger(enterKeyEvent);

// Custom event
$(window).on('startPlay', function(event) {
    console.log('startPlay', event.percent);
});
// Later...
```

```
$(window).trigger('startPlay');
$(window).trigger({type: 'startPlay', percent: '10'});

// OPTIMISATION !

// Listen to events from a particular context (Delegation)
$('table#data').on('click', 'td', function(){
    $(this).toggleClass("active");
});

// Or...
$('table#data').on('click', function() {
    var $cell = $(this).closest('td');
});

// Advanced mousemove
var xCoord = 0,
    yCoord = 0;

$(document).mousemove(function(event){
     xCoord = event.pageX;
     yCoord = event.pageY;
});

var bigFunction = function() {
    // do a lot of things..
    movieParam.left = xCoord;
    //...
    setTimeout(bigFunction, 1000 / 25); // Se rappelle 25/sec
};

bigFunction(); // Init
```

Nb : **Delegated events** have the advantage that they can process events from descendant elements that are added to the document at a later time : it's **LIVE**.

Event Properties

- http://api.jquery.com/category/events/event-object/
- https://developer.mozilla.org/en/docs/DOM/element.addEventListener

jQuery's event system normalizes the event object according to W3C standards. **The event object** is guaranteed to be passed to the **event handler.**
jQuery normalizes properties for cross-browser consistency.
Some properties to remind... :

- **event.currentTarget**
  The current DOM element within the event bubbling phase.
- **event.target**
  The DOM element that initiated the event.
- **event.pageX**
  The mouse position relative to the left edge of the document.

- **event.pageY**
  The mouse position relative to the top edge of the document.
- **event.preventDefault()**
  If this method is called, the default action of the event will not be triggered.
- **event.stopPropagation()**
  Prevents the event from bubbling up the DOM tree, preventing any parent handlers from being notified of the event.
- **event.type**
  Describes the nature of the event.
- **event.which**
  For key or mouse events, this property indicates the specific key or button that was pressed.

event.preventDefault() ?

By default, most events bubble up from the original event target to the document element. At each element along the way, jQuery calls any matching event handlers that have been attached. **A handler can prevent the event from bubbling further up the document tree** (and thus prevent handlers on those elements from running) by calling *event.stopPropagation()*. **Any other handlers attached on the current element will run however**. To prevent that, call *event.stopImmediatePropagation()*. (Event handlers bound to an element are called in the same order that they were bound.)
Similarly, a handler can call *event.preventDefault()* **to cancel any default action that the browser may have for this event**; for example, the default action on a click event is to follow the link. Not all browser events have default actions, and not all default actions can be canceled. See the W3C Events Specification for details.

- http://css-tricks.com/return-false-and-prevent-default/

```
$("a").click(function(event) {
    event.preventDefault();
    event.stopPropagation();
    $("body").append($(this).attr("href"));
}

// IS EQUAL TO

$("a").click(function() {
    $("body").append($(this).attr("href"));
    return false;
}
```

A voir ensemble...

- BINDABLES EVENTS
- KEYS
- Mouse event
- Wait finish // Defered Done
- Gesture (Managing touch event)

- https://github.com/madrobby/keymaster

Exercice et exemples...

1. http://jsfiddle.net/molokoloco/QYYLj/
2. http://jsfiddle.net/molokoloco/hgXyq/

// How to listen the mouse move only when the mouse is down ?

```
var mouseIsMove = function(evt) { console.log(evt.pageX, evt.pageY); };
var mouseIsDown = function(evt) { /* ? */ };
var mouseIsUp = function(evt) { /* ? */ };
$('body')
        .bind('mousedown', mouseIsDown)
        .bind('mouseup click', mouseIsUp);
```

Animations (“Effects”)

Perform a custom animation of a set of CSS properties.

```
$element.animate( properties [, duration ] [, easing ] [, complete ] )
```

- http://api.jquery.com/animate/

➡ jQuery Animate !

```
// Short !
$('p').animate({width:'70%'}, 1500);

$('p').animate({left:'+=50px'}, 1500);

// Semi-Short !
$('p').animate({width:'70%'}, 1500, function() { alert('done'); });

// Full function...
$('#myDiv').animate({    // What we are animating
        opacity:0,
        width:'toggle',
        height:150
    }, {                 // New object with all animation parameters
        duration: 2000,  // how fast we are animating or in seconds
        easing: 'swing', // the type of easing
        specialEasing: {
            width: 'linear',
            height: 'easeOutBounce'
        },
        step: function(now, fx) { // Call for each step of the animation
            var data = fx.elem.id + ' ' + fx.prop + ': ' + now;
            $('body').append('<div>' + data + '</div>');
```

```
            }
            complete: function() { // The callback when done...
                alert('done');
                $(this).addClass('done');
            }
        }
    );

    // Some other shortcut...
    $('div#mydiv').fadeIn(400);
```

➡ Animation Properties and Values

All animated properties should be animated to a single numeric value, except as noted below; most properties that are non-numeric cannot be animated using basic jQuery functionality (**For example, width, height, or left can be animated but background-color cannot be, unless the jQuery.Color() plugin is used**). Property values are treated as a number of **pixels** unless otherwise specified. The units em and % can be specified where applicable.
In addition to style properties, some non-style properties such as scrollTop and scrollLeft, as well as custom properties, can be animated.

Shorthand CSS properties (e.g. font, background, border) are not fully supported. For example, if you want to animate the rendered **border width**, at least a border style and border width other than "auto" must be set in advance. Or, if you want to animate font size, you would use **fontSize** or the CSS equivalent 'font-size' rather than simply 'font'.
In addition to numeric values, **each property can take the strings 'show', 'hide', and 'toggle'**. These shortcuts allow for custom hiding and showing animations that take into account the display type of the element.

Animated properties can also be relative. If a value is supplied with a leading += or -= sequence of characters, then the target value is computed by adding or subtracting the given number from the current value of the property.

Attention, il faut favoriser les animations en CSS

D'une manière général il faut privilégier les animations en CSS (3)
Elles sont plus rapides car effectuées nativement par le navigateur (hardware acceleration)

- Ex. : http://jsfiddle.net/molokoloco/C3TVg/

Pour les animations, le JavaScript peut avoir à faire trop d'appels et de mises à jour sur le DOM.
De plus, les calculs sont effectués entièrement par le processeur.

Callback "complete" :

Il est très pratique de pouvoir attendre la fin d'une animation avant d'effectuer une nouvelle action...

Exemples d'utilisations...

```
// Animate Shortcut : shorcut(fctComplete)
$('p').fadeOut(function() { alert('done'); });

// Animate : animate(properties, duration, fctComplete)
$('p').animate({width:'70%'}, 1500, function() { alert('done'); });

// More params ?... animate(properties, objParams)
$('#myDiv').animate(
      {opacity:0},
      {easing: 'swing', complete: function() { alert('done'); }}
);
```

Entre deux animations il est aussi possible de placer une fonction dans la "**queue**" et d'avoir ainsi l'équivalent d'un callback "complete" sur l'animation...

- http://api.jquery.com/queue/

  *Every element **can have one to many queues of functions** attached to it by jQuery. In most applications, only one queue (called **fx**) is used. **Queues allow a sequence of actions to be called on an element asynchronously, without halting program execution**. The typical example of this is calling multiple animation methods on an element. For example:*

  ```
  $("div").slideUp().fadeIn();
  ```

  *When this statement is executed, the element begins its **sliding animation immediately, but the fading transition is placed on the fx queue to be called only once the sliding transition is complete**.*

Exemples de l'utilisation de "queue()" :

```
$("div").show("slow");
$("div").animate({left:'+=200'}, 2000);
$("div").queue(function () {
    $(this).addClass("newcolor"); // AddClass when animate left end
    $(this).dequeue(); // Continue queue
});
$("div").slideUp();
```

Que l'on peut résumer...

```
$("div")
    .show("slow")
    .animate({left:'+=200'}, 2000)
    .queue(function () {
        $(this)
            .addClass("newcolor")
            .dequeue(); // Continue queue
    })
    .slideUp();
```

A noter, il est aussi possible d'utiliser le paramètre "queue" de la méthode "animate()"

- **queue** (default: true)
  - Type: Boolean
  - A Boolean indicating whether to place the animation in the effects queue. If false, the animation will begin immediately.

```
$('div').fadeIn();
$('div').animate(
      {width:'100%'},
      {queue: false, duration: 1000}
);
```

Méthode "stop(true, false);"

"Stop the currently-running animation on the matched elements"

- http://api.jquery.com/stop/
- http://css-tricks.com/examples/jQueryStop/
- Exemple live : http://jsfiddle.net/molokoloco/P43FG/

Exercices :

- Playground :http://jsfiddle.net/molokoloco/6GGqb/
- Playground2 : http://jsfiddle.net/molokoloco/8t6Qk/
- Doc : http://api.jquery.com/animate/

Exemple de combinaison avec window.setTimeout() et jQuery animate()

```
$(function() {

    var $container = $('#boxFxZone1');

    var makeDivApper = function($div_) {
        $div_
            .appendTo($container)
            .slideUp(0)
            .delay(800)
            .fadeIn(400);
    };

    $.ajax({
        cache: false,
        dataType: 'jsonp',
        url: 'http://search.twitter.com/search.json',
        data: {q: 'html5'},
        success: function(data) {
            $.each(data.results, function(index) {
```

```
                    var tweet = data.results[index];

                    var $div = $('<div><img
src="'+tweet.profile_image_url+'"><strong>@'+tweet.from_user_name+'</strong>'+tweet
.text+'</div>&#8203;');

                    setTimeout(makeDivApper, 1000 * index, $div);
                });
            }
        });

});
```

Les limites...

Ce n'est pas la timeline de Flash !
Difficultées à gérer de longues séquence d'animations sans retour visuel ni interface graphique...

Framework JS qui montre une piste pour combler ce déficite :

- https://github.com/vorg/timeline.js

IDE générants du code JS :

- http://www.sencha.com/products/animator
- http://html.adobe.com/edge/animate/

Une parenthèse sur Bézier et les "easings"...

- http://jqueryui.com/demos/effect/easing.html
- http://www.robertpenner.com/easing/easing_demo.html
- http://cubic-bezier.com
- http://janne.aukia.com/easie/

Exercice :

- http://addyosmani.com/resources/zoominfo/index.html

...

Further READING about CSS3 and animations ? (You would ;)

- **Docs**
  - https://developer.mozilla.org/en/CSS/CSS_animations
  - http://developer.apple.com/library/safari/#documentation/InternetWeb/Conceptual/SafariVisualEffectsProgGuide/
  - http://www.w3.org/TR/css3-3d-transforms/#transform-functions
  - http://www.w3schools.com/css3/css3_animations.asp
  - http://lea.verou.me/2011/10/animatable-a-css-transitions-gallery/
  - http://instacss.com/

    - http://webdesign.tutsplus.com/tutorials/htmlcss-tutorials/css3-transitions-and-transforms-from-scratch/
- **Plugins**
    - https://github.com/codler/jQuery-Css3-Finalize
    - http://ricostacruz.com/jquery.transit/
    - https://github.com/benbarnett/jQuery-Animate-Enhanced
    - http://coding.smashingmagazine.com/2011/11/21/create-web-animations-with-paperjs/
    - http://marcinignac.com/blog/timeline-js/
    - https://github.com/Darsain/motio
    - http://lesscss.org/
    - http://www.queness.com/post/9999/10-css-and-javascript-animation-plugins-frameworks-and-libraries
- **Examples**
    - https://gist.github.com/984039
    - http://lea.verou.me/2011/09/a-better-tool-for-cubic-bezier-easing/
    - http://lea.verou.me/2011/09/pure-css3-typing-animation-with-steps/
    - http://jsfiddle.net/leaverou/7rnQP/light/
    - http://jsperf.com/class-vs-style
    - http://jsfiddle.net/molokoloco/rf8zt/ (CSS animation(s) with 'options.keyframes' OBJ)
    - http://jsfiddle.net/molokoloco/7rV7a/ (CSS 3D animations via sources)

Les Formulaires

Rappel :
Les éléments de formulaires :

- **http://nativeformelements.com**
    - form, input, textarea, select, button, label, ...

Exemple complet (et minimaliste) de l'envoi d'un formulaire via jQuery

- http://jsfiddle.net/molokoloco/CkLA7/

```
<!DOCTYPE html>
<html>
<head>
    <style>
        p { margin:0; }
        div, p { margin-left:10px; }
        span { color:red; }
    </style>
</head>
<body>
    <p>Type 'correct' to validate.</p>
    <p><span></span></p>

    <form action="update.php" method="get">
```

```
            <div>
                <input type="text" />
                <input type="submit" />
            </div>
        </form>

        <script src="http://code.jquery.com/jquery-latest.js"></script>
        <script>
            $("form").submit(function () {
                if ($("input:first").val() == "correct") {
                    $("span").text("Validated...").show();
                    return true;
                }
                $("span").text("Not valid!").show().fadeOut(1000);
                return false;
            });
        </script>
    </body>
    </html>
```

➡ $formElement.val()

Pour obtenir la valeur d'un champs de formulaire (saisie par le visiteur), une seule méthode jQuery !

- http://api.jquery.com/val/

```
var name  = $("input#name").val();
var price = $("select#priceList").val();
var infos = $("textarea#infos").val();
// ...
```

Form serialize ?

The .serialize() method creates a text string in standard URL-encoded notation.
It can act on a jQuery object that has selected individual form controls, such as <input>, <textarea>, and <select>: $("input, textarea, select").serialize();

It is typically easier, however, to select the <form> itself for serialization:

```
$( "form" ).on( "submit", function( event ) {
    event.preventDefault();
    console.log( $(this).serialize() ); // name=Toto&age=10&newsletter=1
});
```

In this case, jQuery serializes the successful controls within the form

- http://api.jquery.com/serialize/

Elements properties VS attributes :

- http://api.jquery.com/attr/
- http://api.jquery.com/prop/

The difference between attributes and properties can be important in specific situations. As of jQuery 1.6, the .prop() method provides a way to explicitly retrieve property values, while .attr() retrieves attributes.
For example, selectedIndex, tagName, nodeName, nodeType, ownerDocument, defaultChecked, and defaultSelected should be retrieved and set with the .prop() method.

Concerning boolean attributes, consider a DOM element defined by the HTML markup <input type="checkbox" checked="checked" />, and assume it is in a JavaScript variable named elem:

```
elem.checked              true (Boolean) // Will change with checkbox state
$(elem).prop("checked")   true (Boolean) // Will change with checkbox state
$(elem).attr("checked")   "checked" (String) // Initial state of the checkbox; does
not change
```

According to the W3C forms specification, **the checked attribute is a boolean attribute, which means the corresponding property is true if the attribute is present** at all—even if, for example, the attribute has no value or is set to empty string value or even "false". This is true of all boolean attributes.

Nevertheless, the most important concept to remember about the checked attribute is that it does not correspond to the checked property. **The attribute actually corresponds to the defaultChecked property and should be used only to set the initial value of the checkbox. The checked attribute value does not change with the state of the checkbox, while the checked property does.**
Therefore, the cross-browser-compatible way to determine if a checkbox is checked is to use the property:

```
if ( elem.checked )
if ( $(elem).prop("checked") )
if ( $(elem).is(":checked") )
```

The same is true for other dynamic attributes, such as selected and value.

➡ Un plugin pour la validation de formulaire !

Pour **une gestion propre des formulaires**, on utilise fréquement le plugin **“jquery.validate.js”**

- http://docs.jquery.com/Plugins/Validation
- http://bassistance.de/jquery-plugins/jquery-plugin-validation/
    - **http://jquery.bassistance.de/validate/demo/**

Il est aussi conseillé d’utiliser un framework CSS avec gestion des formulaires et des erreurs

- **http://twitter.github.com/bootstrap/base-css.html#forms**
- Démo : http://alittlecode.com/jquery-form-validation-with-styles-from-twitter-bootstrap/

Exercices :

1 / Customiser l'exemple suivant :
Créer un formulaire de contact : Nom et prénom, email et message...

- http://jsfiddle.net/molokoloco/CkLA7/

Tester que tous les champs sont présents et qu'un arrobase ("@") existe dans l'adresse email, avant de valider le tout vers le backend de votre site, un fichier "actions.php" dans le répertoire imaginaire "php/" à la racine de votre site

2/ Télécharger le site avec un formulaire jQuery complet (JS + PHP) ci-joint :
Modifier le front office et le back office (php) pour l'adapter à ce nouveau format de formulaire :
Nom et prénom, email et message... plus "checkbox" "S'inscrire à la newsletter ?"

- https://dl.dropbox.com/u/17715374/site.form.contact.jquery.zip

Les requêtes "AJAX"

➡ **AJAX** : Principe premier, **intérroger un fichier texte (string) externe à la page**, de manière **asynchrone**, et ensuite traiter le résultat en JavaScript et eventuellement l'afficher dans une partie de la page actuelle.
Les données utilisées pour faire de l'Ajax sont (souvent) :

- **Une page HTML/PHP (du même site),**
- **Un fichier JSON**
- **Un fichier XML**
- **Ou encore un simple fichier texte .TXT, JavaScript .JS …**

Dans certains cas précis on peut faire une requête AJAX qui soit **synchrone**, c'est à dire **qui bloque l'execution du code JS** tant que le resultat n'est pas obtenu (Mais c'est moche !)
On change parfois **la méthode d'appel GET** (par défaut) par POST pour envoyer plus de données sur un serveur ou protéger ces variables (pas de "url encoding").

GET et POST font tous les deux une requête vers le serveur
et obtiennent en retour une réponse de celui-ci.

- http://www.diffen.com/difference/Get_vs_Post
  - **GET :** Transmission des variables par URL
  - **POST :** Transmission des variables dans le header de la page

Avec jQuery ajax, on peut abstraire les données et les rendre agnostiques à la méthode GET ou POST

```
// Work...
$.ajax({url:'./backend.php?param=hello&q=2'});

// Do not Work
$.ajax({method:'post', url:'./backend.php?param=hello&q=2'});

// Work well !
$.ajax({method:'get', url:'./backend.php', data:{param:'hello', q:2} });
$.ajax({method:'post', url:'./backend.php', data:{param:'hello', q:2} });
```

Formats des données demandées/reçues

- http://api.jquery.com/jQuery.ajax/

➡ **jQuery peut prendre un paramètre "dataType" en entrée (par défaut "Intelligent Guess") :**

- **dataType** (default: Intelligent Guess (xml, json, script, or html))
  - Type: String
  - The type of data that you're expecting back from the server. If none is specified, jQuery will try to infer it based on the MIME type of the response (an XML MIME type will yield XML, in 1.4 JSON will yield a JavaScript object, in 1.4 script will execute the script, and anything else will be returned as a string). The available types (and the result passed as the first argument to your success callback) are:
    - **"xml":** Returns a XML document that can be processed via jQuery.
    - **"html":** Returns HTML as plain text; included script tags are evaluated when inserted in the DOM.
    - **"script":** Evaluates the response as JavaScript and returns it as plain text. Disables caching by appending a query string parameter, "_=[TIMESTAMP]", to the URL unless the cache option is set to true. Note: This will turn POSTs into GETs for remote-domain requests.
    - **"json":** Evaluates the response as JSON and returns a JavaScript object. The JSON data is parsed in a strict manner; any malformed JSON is rejected and a parse error is thrown. As of jQuery 1.9, an empty response is also rejected; the server should return a response of null or {} instead. (See json.org for more information on proper JSON formatting.)
    - **"jsonp":** Loads in a JSON block using JSONP. Adds an extra "?callback=?" to the end of your URL to specify the callback. Disables caching by appending a query string parameter, "_=[TIMESTAMP]", to the URL unless the cache option is set totrue.
    - **"text":** A plain text string.
    - **multiple, space-separated values:** As of jQuery 1.5, jQuery can convert a dataType from what it received in the Content-Type header to what you require. For example, if you want a text response to be treated as XML, use "text xml" for the dataType. You can also make a JSONP request, have it received as text, and interpreted by jQuery as XML: "jsonp text xml." Similarly, a shorthand string such as "jsonp xml" will first attempt to convert from jsonp to xml, and, failing that, convert from jsonp to text, and then from text to xml.

En retour, sur le callback “success”, on recoit trois arguments :

- **success**
  - Type: Function(PlainObject data, String textStatus, jqXHR jqXHR)
  - A function to be called if the request succeeds. The function gets passed three arguments: **The data returned from the server, formatted according to the dataType parameter;** a string describing the status; and the jqXHR object

➡ JSON encode / Decode ?

Native JSON.stringify (Missing in jQuery) and JSON.parse (“**$.parseJSON()**”) :

- **http://api.jquery.com/jQuery.parseJSON/**
- http://stackoverflow.com/questions/191881/serializing-to-json-in-jquery

```
// String to JSON with jQuery (better)
var obj = $.parseJSON('{"name":"John"}');
console.log(obj.name === "John");

// String to JSON with native JS on moderns browsers
var obj = JSON.parse('{"name":"John"}');
console.log(obj.name === "John");

// JSON to String
var myObj = {foo: "bar", "baz": "wockaflockafliz"};
console.log(JSON.stringify(myObj));
// '{"foo":"bar","baz":"wockaflockafliz"}'
```

“*JSON.stringify*”, n’est que nouvellement implémenté sur les navigateurs.
Pour le moment il vaut mieux donc utiliser un plugin “**jquery.json.js**” pour convertir/exporter les objets JSON sous forme de chaine de caractères.

La version complete du hack est disponible ici :

- https://code.google.com/p/jquery-json/source/browse/trunk/src/jquery.json.js
- Cf. http://stackoverflow.com/questions/3904269/convert-object-to-json-string

When working with JSON parsing it’s better to wrap with “try catch”...

```
try {
    // prefer jQuery instead of window.JSON.parse()
    var liveOptions = $.parseJSON(val);
}
catch(e) {
    console.log('JSON Parse error');
}
```

➡ Interne au site (même domaine)

La fonctions principale pour faire de l'AJAX est : **$.ajax()**

- http://api.jquery.com/category/ajax/
- http://api.jquery.com/category/ajax/shorthand-methods/

jQuery prévoit de **nombreuse variantes** qui permettent de ne pas spécifier tous les paramètres de cette fonction. Par exemple, la plus simple :

```
$.ajax({
    url: './data/test.html', // Simple fichier HTML, sans head ni body
    success: function(data) {
        $('.result').html(data);
    }
});
```

...une autre equivalente :

```
$('<div id="#slide-2"></div>')
    .load('slides/2.html')
    .appendTo('body');
```

Avec quelques paramètres..

```
$.ajax({
    type: 'POST',
    url: './backend.php',
    data: {param:'hello'}, // Envoyer un paramètre avec la méthode "POST"
    complete: function(xhr, textStatus) {
        $ui.removeLoader();
    },
    success: function(data) {
        console.log(data);
        $('#content').fadeOut('slow',function(){
            $(this).html(data.template).fadeIn();
        });
    },
    error: function(XMLHttpRequest, textStatus, errorThrown) {
            console.log('JSON error :(', XMLHttpRequest, textStatus, errorThrown);
    }
});
```

… et un de ces “shortcuts” équivalent :

```
$.post({
    url: './backend.php',
    data: {param:'hello'},
});
```

Externe au site (Cross-domain)

Pour faire des requêtes externes à son propre domaine,
en JavaScript, il faut utiliser la méthode JSONP

http://blog.jaysalvat.com/articles/comprendre-jsonp-et-acces-de-donnees-a-distance-en-javascript.php

En effet, **pour des raison de sécurité, un site ne peut pas aller prendre les ressources d'un autre site sans technologie assimilable à un "proxy"** (Herbergé sur le serveur de la page actuelle), à noter les **Websockets** HTML5 et les hacks via **iframe**, qui sont encore différents...

L'unique moyen pour JS d'ajouter une nouvelle donnée en provenance d'un site extérieur dans une page est d'ajouter dynamiquement un nouveau fichier JS :

```
var loadJs = function(src) {
    $('head').append('<' + 'script type="text/javascript" src="'+src+'"></sc'+'ript>');
};
```

Le fichier JS linké sera évalué par le navigateur comme un fichier dynamique JavaScript et évalué comme tel.
Si le fichier renvoie :

```
moncallback({"data":"Plein de donnés utiles"})
```

Alors le navigateur appel la fonction pré-existante "moncallback()" en lui passant la chaine JSON en argument. Les données JSON passées en argument seront traitées dans cette fonctions à vous.

jQuery simplifie la tâche et redirige l'appel de la fonction de callback JSONP sur le callback "success" de votre requête $.ajax()
Il génère donc un nom de callback aléatoire qu'il passe en argument (GET) et que votre fichier PHP doit prendre en compte.

```
[                          Vous                          ][        jQuery            ]
http://www.sitedistant.com/api.php?action=getclient&callback=randomCallBack0123
```

Le fichier renvoyé par le PHP doit donc contenir une chaine du type :

```
randomCallBack0123({"foo":"bar"})
```

Il sera exécutée dans le contexte de la page actuelle comme étant du JavaScript.

Exemple de page PHP hébergée sur un autre domaine :
http://www.b2bweb.fr/_COURS/api.php

```
<?php
// Array PHP contenant les infos à renvoyer
$customData = array('key1' => 'Plein de données utiles');

// Stringified to JSON
$jsonData = json_encode($customData); // '{"key1":"Plein de donnés utiles"}'
```

```php
// jQuery JSONP params ?
$callBack = (isset($_GET['callback']) ? clean($_GET['callback']) : NULL);

// Do the response..
if ($callBack) {
    header('Content-type: application/json');
    echo $callBack.'('.$jsonData.')';
    //=> moncallback({"key1":"Plein de donnés utiles"})
}
else {
    header('Content-type: text/json');
    echo $jsonData;
    //=> {"key1":"Plein de donnés utiles"}
}
?>
```

Exemple de JavaScript à utiliser dans la page actuelle :
http://www.monsite.com/js/main.js

```javascript
$(function() {
      $.ajax({
            dataType:'jsonp',
            url: 'http://www.b2bweb.fr/_COURS/api.php',
            data: {action:'getUser'},
            success: function(data) {
                console.log(data);
                // "Plein de donnés utiles"
            },
            error: function(XMLHttpRequest, textStatus, errorThrown) {
                console.log('JSON error', XMLHttpRequest, textStatus, errorThrown);
            }
      });
});
```

...Ou encore... (etc...)

```javascript
var getLatestFlickrPics = function(tag, callback) {
    var flickrFeed =
'http://api.flickr.com/services/feeds/photos_public.gne?tags='+ tag +
'&tagmode=any&format=json&jsoncallback=?';
    $.getJSON(flickrFeed, callback);
};

// Usage :
getLatestFlickrPics('ferrari', function(data){
    $.each(data.items, function(i, item){
        $('<img/>').attr('src', item.media.m).appendTo('body');
    });
});
```

Exercice : Afficher un stream twitter...

- http://jsfiddle.net/molokoloco/PkxRP/

- http://jsfiddle.net/molokoloco/6GGqb/

Pour aller plus loin

- Utilisation de **Templates** HTML mappés sur des **objets JSON**
  - http://jsfiddle.net/molokoloco/wtHeX/ (Avec jquery.tmpl)
  - http://jsfiddle.net/molokoloco/RgAhT/ (Avec Moustache.js)

“Promise” et “Deferred” ?

Introduction...

The "Deferred" pattern describes an object which acts as a proxy for some unit of computation that may or may not have completed. ***The pattern can apply to any asynchronous process: AJAX requests, animations, or web workers to name a few****. Even user behavior can be thought of as a "delayed computation."*

“As asynchronous operations run in background and we can’t determine when they will complete*, their failures cannot be caught by the catch block. Even in case of successful completion, the response won’t be available for the immediate next statement. These cases are handled using callbacks”*

jQuery “Deferred” ?

- http://api.jquery.com/jQuery.Deferred/
- http://api.jquery.com/category/deferred-object/

“a chainable utility object with methods to register multiple callbacks into callback queues*, invoke callback queues, and relay the success or failure state of any synchronous or asynchronous function”*

```
// Example with the “$.when().done()” pattern...
var effect = function() {
   return $('div').fadeIn(800).delay(1200).fadeOut();
};

$('button').on('click', function() {
   $('p').append('Started...');
   $.when( effect() ).done(function() {
       $('p').append('Finished!');
   });
});

// Example with only the “done” callback...
$.get("test.php").done(function() {
    alert("$.get succeeded");
```

```
});
```

Pour que ceci marche, les objets **$.ajax()** et **$.animate()** héritent déjà des propriétés de Deferred (Ainsi que leur shortcut)

```
// Assign handlers immediately after making the request,
// and remember the jqxhr object for this request
var jqxhr = $.ajax( "example.php" )
   .done(function() { alert("success"); })
   .fail(function() { alert("error"); })
   .always(function() { alert("complete"); });
// perform other work here ...
// Set another completion function for the request above
jqxhr.always(function() { alert("second complete"); });
```

Understanding Promise Pattern

With multiple callback, promises can mitigate the “Pyramid of Doom”: the situation where code marches to the right faster than it marches forward.

```
step1(function (value1) {
    step2(value1, function(value2) {
        step3(value2, function(value3) {
            step4(value3, function(value4) {
                // Do something with value4
            });
        });
    });
});
```

With a promise library, you can flatten the pyramid.

```
Q.fcall(step1)
.then(step2)
.then(step3)
.then(step4)
.then(function (value4) {
    // Do something with value4
}, function (error) {
    // Handle any error from step1 through step4
})
.done();
```

With this approach, you also get **implicit error propagation**, just like try, catch, andfinally. An error in step1 will flow all the way to step5, where it’s caught and handled.

- http://joseoncode.com/2011/09/26/a-walkthrough-jquery-deferred-and-promise/
- http://sravi-kiran.blogspot.in/2013/03/UnderstandingPromisePatternInJavaScriptUsingjQueryDeferredObject.html
- http://api.jquery.com/category/deferred-object/
- Une autre approche Q.js : http://documentup.com/kriskowal/q/

Exemple d'utilisation de l'objet jQuery "**Deferred**"

Fetch some tweets with Ajax and customise the "$.Deferred()" object...

- **http://jsfiddle.net/molokoloco/2ZmjC/**

Les bases d'un plugin jQuery

Introduction...

"As a developer your job is to absorb complexity and expose simplicity.
End users should reap the benefits of your understanding."

> ***So you've become comfortable with jQuery*** *and would like to learn how to write your own plugins. Great! You're in the right spot.* ***Extending jQuery with plugins and methods is very powerful and can save you and your peers a lot of development time by abstracting your most clever functions into plugins****.*

Pourquoi fait-on un un plugin jQuery ?

Pour créer une fonction capable de manipuler du DOM, des images, de trier des informations en provenance d'une API RESTFULL, de flux RSS, etc...
Nous avons vu qu'il existe par exemple un plugin "*jquery.cookie.js*" : Il n'y a pas un réel intérêt à faire ce genre de plugin qui n'utilise aucune fonction implémentée par jQuery. **L'unique intérêt est alors de l'intégrer mieux dans un projet existant** et qui requière déjà de nombreux autres plugins jQuery (Consistance du code et nameSpacing).

- **Documentation : http://docs.jquery.com/Plugins/Authoring**

Pour des composants d'interface plus avancés, le développeur pourra aussi s'intéresser à la **Widget Factory** de jQuery UI :

- **http://jqueryui.com/widget/**
  Create stateful jQuery plugins using the same abstraction as all jQuery UI widget

Pour des applications, le développeur se tournera vers l'élaboration d'un "Framework", c'est à dire un ensemble de fichiers rassemblés sous une architecture précise et souple et qui comporte des ensembles

de bibliothèques de fonctions, de class, et de plugins dédiés...

➡ Le plus petit plugin jQuery du monde ?

```
(function($){                          // Closure pour protéger ses variables

    // Plugin "setHeight()"
    $.fn.setHeight = function(h) { // Ici le nom du plugin ("$.fn.xxx")
        this.each(function() {     // Boucle sur la collection jQuery
            $(this).height(h);     // Ici le code "custom"
        });
        return this;               // Retour du contexte
    };

})(jQuery);

$('div').setHeight('30px'); // Set the height on a collection of elements
```

Test en live !

Testable jQuery plugin boilerplate :

- http://jsfiddle.net/molokoloco/DzYdE/

Autres exemples...

- https://github.com/molokoloco/FRAMEWORK/tree/master/jquery.plugins

Plugin "Authoring"

1. Getting Started
2. Context
3. The Basics
4. Maintaining Chainability
5. Defaults and Options
6. Namespacing
7. Summary and Best Practices

Aller plus loin...

- jQuery Plugins Patterns Boilerplates book and code :
    - https://github.com/jquery-boilerplate/patterns/tree/master/patterns
    - http://addyosmani.com/resources/essentialjsdesignpatterns/book/#designpatternsjquery
- http://coding.smashingmagazine.com/2011/10/11/essential-jquery-plugin-patterns/
- http://kenneth.kufluk.com/blog/2010/08/chaining-asynchronous-methods-in-jquery-using-the-queue/

Exercices !

A) Récupérer le code du “plus petit plugin jQuery du monde”

1. L’insérer dans un JS fiddle
2. Le modifier pour créer un plugin $.placeBottom
3. Ce plugin doit prendre une collection d’éléments et les placer en position absolue en bas de la fenêtre
4. Le plugin doit calculer la hauteur de la fenetre, la hauteur d’un élément et modifier le css “top” de celui-ci afin de placer le bas de l’élément en bas de la fenêtre
5. Modifier le plugin et le nommer “place”. Il devra maintenant prendre en compte un argument “pos” qui pourra être “left, top, bottom ou right”
6. Placer l’element en fonction dans la fenetre

B) L’exercice est éxécuté avec la méthode JSONP (Cf. chapitre “Requêtes Ajax JSONP”) car le fichier PHP d’exemple est hébergé sur un domaine distant.

- http://www.b2bweb.fr/_COURS/api.php

Etendre le plugin suivant

- http://jsfiddle.net/molokoloco/DzYdE/

Afin que les éléments avec un attribut du type “data-title” déclanchent un appel ajax

```
<a href="#" data-src=”http://www.b2bweb.fr/_COURS/api.php?id=link1”>Tooltip 1</a>
```

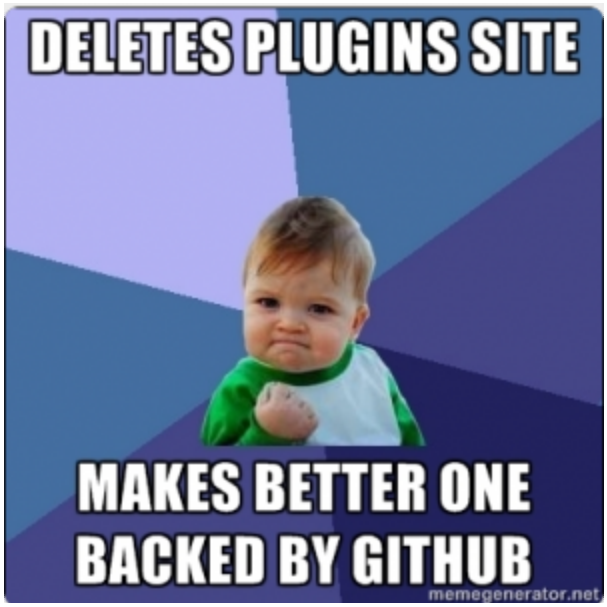
Ce lien devra appeler le fichier externe et afficher un tooltip avec la valeur de la réponse Ajax

1. Dans la méthode “init()”, vérifier la présence de l’attribut “data-src” avec “data()”
2. **Si il y a une valeur, faire un appel “ajax()”**
3. Quand la réponse ajax est reçu (callback “success”)
    a. Essayer de mettre à jour le tooltip en appelant la méthode update() :
    `publicMethods.update(data.key1);`
    Il y a t-il une erreur ?
4. Paramétrer l’appel Ajax pour qu’il garde le **contexte** actuel du “this” au moment de l’appel :
    a. C’est le “this” qui correspond à un élément
    `context: $this`
    (cf. http://api.jquery.com/jQuery.ajax/)
5. Essayer de faire une update de cette manière :
    a. `$.proxy(publicMethods.update, this)(data.key1);`
    Il faut utiliser “$.proxy()” pour appeler la méthode “update()” avec le context “this”
    C’est le “this” qui correspond à un élément (passé en argument de “$.ajax()” : $this)
6. Oui tout ça est un peu compliqué, c’est pour ça qu’il y a un prof !

7. Je veux entendre vos claviers et voir la console ouverte !
8. Ca marche ?
9. Essayez maintenant de déplacer l'appel Ajax dans la méthode show() (pour ne faire l'appel Ajax que lorsque l'utilisateur veux afficher le tooltip)
    a. Pensez à stocker l'url dans les data afin de pouvoir le reutiliser dans la méthode show()

Les plugins jQuery !

WHAT IS HAPPENING TO THE JQUERY PLUGINS SITE ?



- ~~http://blog.jquery.com/2011/12/08/what-is-happening-to-the-jquery-plugins-site/~~
- ~~>>>~~ ~~http://archive.plugins.jquery.com/~~

Plugins site is up and shiny !!!

- **http://plugins.jquery.com**
- **https://github.com/jquery/plugins.jquery.com#readme**

Autres exemples... au hasard...

- **http://jquerypp.com**
- https://github.com/codler/jQuery-Css3-Finalize
- http://ricostacruz.com/jquery.transit/
- http://api.jquery.com/jquery.tmpl/

Pensez à suivre les gens et frameworks intéressants sur **Github** !!
Exemple : https://github.com/molokoloco/following

Ressources for building the new web ^^ (2012)
http://www.b2bweb.fr/molokoloco/best-must-know-ressources-for-building-the-new-web-%E2%98%85/

JSfiddle :

Bon, ok c'est les plugins du prof, mais au moins il sait les expliquer ^^

- Tooltips, un plugin "simple" : http://jsfiddle.net/molokoloco/DzYdE/
- jQuery.viewport.js : http://jsfiddle.net/molokoloco/Atj8Z/
- jQuery & CSS3 analogue clock V2.1 : http://jsfiddle.net/molokoloco/V2rFN/

Autres exemples :

- http://www.b2bweb.fr/molokoloco/molokoloco-coding-project/
- http://www.b2bweb.fr/molokoloco/dynamisez-vos-pages-web-avec-jquery-boxfx-js/
- http://www.b2bweb.fr/bonus/jx/#1
- http://www.b2bweb.fr/molokoloco/jquery-default-plugin/
- http://www.b2bweb.fr/molokoloco/wallify-v-0-8

"My" Wiki jQuery CodeSheat

Vous pouvez toujours y jeter un oeil ;)
http://code.google.com/p/molokoloco-coding-project/wiki/JavascriptJquery
En vrac (depuis 2008 et jusqu'à maintenant...)

- JQUERY TIPS
- LITTLES FUNCTIONS FOR QUICK USE
- (NEW) SAMPLE PLUGIN STRUCTURE
    - 'Highly configurable' mutable plugin boilerplate

Travaux Pratiques

Le TP sur 3 jours...

Travail individuel à rendre (noté sur 20)

Il y a plusieurs thèmes...

- **Navigation onglet** (Tabs)
- **Formulaires** (Form)
- **Carrousel** (Slideshow)
- **Popover** (Tooltip)
- **Popin/modal** (Lightbox)
- **Scrollspy** (Waypoints)

Chacun devra me montrer et m'envoyer un ou plus de ces plugins jQuery, intégré dans une page d'un site à lui (un blog, un faux site, un wordpress....)
Je fournirai une liste mais nous apprendrons aussi à chercher un plugin et l'installer au sein de pages web, pour l'intégrer.
Nous choisirons ensemble un (ou des ?) plugin à construire de zéro et/ou customiser fortement, dans un de ces thèmes, et l'intégrerons dans une architecture complète de site.

Outils à tester pour le cours ?

- http://codeshare.io
- https://towtruck.mozillalabs.com/example/code

Liste de ressources utiles :

Architecture de base d'un site :

- http://html5boilerplate.com/
- https://github.com/h5bp/html5-boilerplate

```
./monsite/
./monsite/css/
./monsite/css/fonts/
./monsite/css/jqueryuiTheme/
./monsite/img/
./monsite/img/slide-accueil/
./monsite/js/
./monsite/js/vendor/
./monsite/js/bootstrap/
```

Code source site d'exemple :

- **https://dl.dropboxusercontent.com/u/17715374/www.siteref.com.zip**

Structure type page index :

- HEAD :
    i. frameworks CSS (Ex. boostrap)
    ii. plugins CSS (jQuery UI)
    iii. Customs style CSS (styles.css)

iv. En option : Modernizr JS (ou HTML5 Shiv)

- BODY :
  a. HTML...
     i. Elements ID, Class, ...
  b. JAVASCRIPT :
     i. jQuery
     ii. Plugin jQuery
     iii. Autre framework optionnels
     iv. Main.js

Structure type d'un plugin :

- return this.each(function() {} // Iterate current(s) element(s) collection
- Merge option par défauts avec options donné par l'utilisateur
- Génére des éléments dynamiquement ?
- Ajoute des écouteurs (Listener) ?
- Renvois des méthodes publiques que l'utilisateur peut appeler sur l'élément ciblé par le plugin ?
- Travail sur des méthode internes "Privates" uniquement utilisables au sein du plugin ?

Twitter Boostrap :
Un framework JS/CSS recouvrant des composants HTML et des plugins JS parmis les plus utilisés :

- http://twitter.github.io/bootstrap/getting-started.html#examples
- https://github.com/twitter/bootstrap
- Twitter Bring Bootstrap's components to life, now with 13 custom jQuery plugins.

jQuery UI :

- jQuery UI is a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library.
- + jQuery UI Bootstrap theme, the incredible styles of Twitter Bootstrap, for jQuery UI

Plugins "individuel" de composants d'interface :

- JQuery Slideshow/Carousel Plugin. Sources.
- Swipebox, A touchable jQuery lightbox
- Photobox A lightweight image gallery modal window script
- Flux Slider Transitions Gallery, hardware accelerated image transitions using CSS3
- Roundabout for jQuery, 3D effect carousel with sprites (Div) that turn around an axe
- Reveal.js The HTML Presentation Framework
- Supersized Full Screen Background/Slideshow jQuery Plugin
- Isotope (Masonry V2), an exquisite jQuery plugin for magical layouts Reveal & hide items with filtering Re-order items with sorting Dynamic, intelligent layouts
- jQuery simple Layout Demo that manage slidable and resizable, multi-pane
- SlickGrid is a JavaScript grid/spreadsheet component

- typeahead.js is a fast and fully-featured autocomplete library
- …
- Customizable **boilerplate/template** of a jQuery plugin, EDIT here !
- …
- **A part :**
    - **jPlayer** : HTML5 Audio & Video for jQuery
    - **SoundManager 2**: JavaScript Sound For The Web
    - ProcessingJS : using <canvas> for a drawing surface
    - … Three.js, Raphael.js, ... etc

Plugins "helpers" sur certaines fonctionnalités

- **jQueryColor**, this plugins installs a cssHook which allows jQuery's .css() and .animate() to animate between two colors. The jQuery.Color() function allows you to create and manipulate color objects or string that are accepted by jQuery's
- jQuery Css3-like **Easings** for jQuery animations
- **jQuery.animate** **Overwrites $.fn.animate to use CSS 3 animations if possible**. It takes the same arguments as the original $.fn.animate and will fall back to jQuery's JavaScript animation if a CSS animation is not possible.
- Alternatively, for somes simples animations, there is animate.css a bunch of cool, fun, and cross-browser CSS3 animations... or jQuery CSS 3 Finalize With this plugin you can write CSS without the vendor prefixes. The plugin takes care of it and will automatically add vendor prefixes.
- jquery.transit Super-smooth CSS transitions & transformations for jQuery
- Spritely is a jQuery plugin for creating dynamic character and background animation (Pan, Scroll, Sprite)
- jQuery Waypoints is a small jQuery plugin that makes it easy to execute a function whenever you scroll to an element with or without infinite scrolling enabled
- jQuery **Form validation** Plugin
- jQuery **Sisyphus** plugin developed to save html forms data to LocalStorage to restore them after browser crashes, tabs closings and other disasters
- jQuery **Remember stuff**. A simple way to set/read/destroy cookies and localstorage
- Scrollorama, The jQuery plugin for doing cool scrolly stuff
- jQuery and Wordpress plugins for infinite scroll
- **Moment.js** Parse, validate, manipulate, and display dates in JavaScript
- **Keymaster** Awesome handling of keyboard events
- jQuery Mouse Wheel Plugin A jQuery plugin that adds cross-browser mouse wheel support...
- Hand.js is a polyfill for supporting pointer events on every browser. Pretty simple: Register pointer events (pointerdown, pointerup, pointermove, etc..). It emulate click, touchstart, etc...
- Pinch, Zoom, Scroll, iScroll-4 !
- Grunt : The JavaScript Task Runner (Compiler)
- …

Useful JavaScript Libraries and jQuery Plugins :
http://coding.smashingmagazine.com/2012/09/23/useful-javascript-libraries-jquery-plugins-web-developers/

- **Web Forms :** forms framework - auto-saving drafts - file upload (and resuming large downloads) - <select> boxes - modal boxes - form accordion - dynamic labels - drop-down with images - tooltips - extended input - form validation - credit card validation(alternative) - email check - password complexity
- **Web Typography :** repairing vertical baseline - align text to a grid - responsive measure - fixing widows - fluid line height - scalable headlines (or smart headlines) - Lettering.js -Kerning.js
- **Little Time-Savers :** exchange rates and currency - date/time formatting - relative timestamps -number and currency formatting - cookies.js - zip.js - extra string methods -countdown.js - sticky content - Google Maps - interactive maps - progress bar -favicon notifications (or Notificon)
- **Images, Maps, Graphs :** world maps - subway map - Google maps - open source maps - SVG fallback - gauges - graphs - timeline - Retina display - magnifying glass - interactive graphs- plots - time visualization
- **Text Manipulation :** syntax highlighter - URI.js - jQuery URL parser - cutting paragraphs - text truncation - TOC generator - FAQ generator - sorting text by relevancy.
- **Manipulating Tables and Lists :** table styling - searchable/sortable lists - visual search - nested sortable lists - large data sets - CSV to table conversion (or Csonv.js) - Excel-like tables - advanced tables
- **Useful JavaScript Tools for Web Development :** Yeoman - command line for JS - image placeholder - percentage loader - URL parser - URI normalization - touch events - multi-touch gestures - Markdown Embedding - accessibility enhancement - templating engine - filepicker -extensible regex - client-side caching
- **HotKeys :**
  https://github.com/jeresig/jquery.hotkeys/blob/master/jquery.hotkeys.js
  https://github.com/madrobby/keymaster
  http://dmauro.github.io/Keypress/

Online tools... (Via http://home.b2bweb.fr/#code)

- **HTML/CSS :** **DocHubNew**, W3cHtml, W3cCss, W3cValidator, HtmlColors, Html5Rocks, **CssBeautifier**, **CssCompressor**, **Css3Maker**, **Css3Gradient**, ModernizR
- **JAVASCRIPT :** **jQueryDoc**, MicroJS, MozillaJS, WindowsJS, ProcessingJS, Three.JS, SanchaDoc, SanchaTouchDoc, PhoneGap, PhpJS, GoogleJsApi, JSperf, MinifyJS, **ClosureCompiler**, **JsBeautifier**, **JsonBeautifier**, Json Parser
- **LAMP :** Apache, **PhpNet**, PhpClasses, Zend, PhpAnywhere, PhpBeautifier, MySqlDoc, SQLite, SqlDesigner, HtAccess, ZoneCheck, Whois, IpTools, PageSpeed, **LocalHost**
- **EDITING :** CodeRun, IdeOne, **JsFiddle**, **Cloud9**, Akshell, Bespin, TinyMce, Balsamiq, Mindomo, Aviary, PixlR, FavIcon
- **DESIGN :** **ColorScheme**, Kuler, CssPrism, IconLook, FreeIcons, IconFinder, MyFontbook, DaFont, Symbols, Typetester, TypeFolly, FontSquirrel, SpecialChar, **GoogleWebFonts**
- **DATA :** WikiMolokoloco, GoogleAPI, ProgrammableWeb, SourceForge, **GitHub**, CheatSheets, YahooDeveloper, FirefoxAddons, ProgrammerTools, PatternTap, YahooPipes
- ...

Exemple de pages et codes source intéressants, à faire (suggestion forte :)

- Image “zoom infos” effect (Niveau noob)
- A Simple Parallax Scrolling Technique (niveau débutant)
- Un exemple d’animation complète du contenu dans une page ? (niveau intermediaire)
- ...

Sites de ressources...

- **Plugins.jQuery** jQuery new plugins center
- **jQuery++** Collection of extremely useful DOM helpers and special events for jQuery 1.8 and later. I’m not a UI project like jQuery UI or jQuery Tools. Instead, I’m all about providing low-level utilities for things that jQuery doesn’t support
- Unheap A tidy repository of jQuery plugins
- jQueryList The Ultimate jQuery List
- Bootswatch Free themes for Twitter Bootstrap...
- ...et toutes les... “listes des 50 meilleurs plugins jQuery” ^^
    - http://coding.smashingmagazine.com/2011/04/07/useful-javascript-and-jquery-tools-libraries-plugins/

ANNEXES

Une liste assez complète des ressources d'apprentissage disponibles ici :
http://www.learningjquery.com/2010/07/great-ways-to-learn-jquery

TUTORIAUX “Officiels” JQUERY

http://docs.jquery.com/Main_Page

GETTING STARTED

- Downloading jQuery
- How jQuery Works
- Frequently Asked Questions
- Tutorials
- Using jQuery with Other Libraries
- Variable Types

PLUGINS

- Plugin Repository
- Plugin Authoring

SUPPORT

- Frequently Asked Questions
- Forums and Chat
- Submit a Bug Report
- Commercial Support

ABOUT JQUERY

- Browser Compatibility

TUTORIAL - NETTUTS

http://www.bennadel.com/blog/1492-An-Intensive-Exploration-Of-jQuery-With-Ben-Nadel-Video-Presentation-.htm

- Introduction
- What Is jQuery
- UI Effects – Pain Free Animation
- Why I Didn't Like jQuery At First
- jQuery For Developers
- Anonymous Methods
- $() Factory Method
- Wrapping DOM Elements
- jQuery Selectors
- jQuery Selector Moment of Bliss
- Working With The $() Collection
- Attributes And Values
- Moving Elements Around
- Traversing The DOM
- Filtering The jQuery Collection
- Iterating Over The Stack
- jQuery Closures – Awesome Voodoo Magic!
- Eventing Binding And Triggering
- Custom Event Types
- jQuery AJAX
- Monitoring AJAX Requests
- jQuery Data() Method
- Extending jQuery – Plugins And Selectors
- jQuery Is Mad Awesome
- jQuery Resources

TUTORIAL - NETTUTS 2

http://learnjquery.tutsplus.com/

- **DOM Traversal** jQuery provides dozens of different methods for traversing the DOM. You'll learn what the DOM is, and how jQuery can be used to fetch elements.
- **Events** We respond to specific actions from the user through the use of events. "When the user clicks, or hovers, or double-clicks...do this."
- **DOM Manipulation** We're not merely limited to retrieving information from the DOM; we can manipulate it as well! I'll teach you how to create, remove, and modify elements within a document.
- **Effects** If executed properly, a touch of animation can go a long way in an application. You'll learn how to use jQuery's animate method to enhance your applications.
- **AJAX** With traditional JavaScript, the process of making asynchronous requests wasn't a job for the faint of heart. Luckily, jQuery makes the task laughably simple.
- **Plugin Development** One of the secrets to jQuery's success is its vibrant plugin development community. You'll learn how to use existing plugins, and create your own from scratch.

TUTORIAL - SITES DU ZERO

http://www.siteduzero.com/tutoriel-3-689034-simplifiez-vos-developpements-javascript-avec-jquery.html

Introduction à jQuery

Avant de commencer
Historique du Web : de HTML à jQuery
Qu'est-ce que jQuery ?
Ce qui rend jQuery si puissant et universel
Installer jQuery
Chapitre 2
Premiers pas
Le vocabulaire à connaître
Le squelette HTML typique
Attendre la disponibilité du DOM
Premier script : « Hello world »

Les bases de jQuery

Chapitre 1
- Sélection d'éléments
- Fonctionnement de base de jQuery
- Sélection d'éléments
- Notions indispensables

Chapitre 2
- Plus loin dans la sélection d'éléments
- Sélecteurs d'attributs

Sélecteurs hiérarchiques
Pseudo-sélecteurs d'éléments sélectionnés
Sélecteurs d'éléments particuliers
Pseudo-sélecteurs spécifiques aux formulaires
Sélecteurs utilisés dans les tableaux
Parcourir les éléments sélectionnés
Conversion jQuery/DOM
Chapitre 3
Modifier le contenu d'un élément
Getters et setters
Accéder aux attributs HTML et aux propriétés CSS
Travailler avec l'attribut class
Travailler avec les formulaires
Travailler avec les valeurs stockées dans des éléments
Position et taille des éléments
Associer des données aux balises
Chapitre 4
Insérer et remplacer des éléments dans le DOM
Insérer du contenu
Remplacer des éléments
Insérer des éléments
Déplacer du contenu
Dupliquer des éléments
Entourer des éléments
Supprimer des éléments
Chapitre 5
TP : Questionnaire interactif en jQuery
Instructions pour réaliser le TP
Correction

Aller plus loin avec jQuery

Chapitre 1
Les bases de la gestion événementielle
La souris
Le clavier
Les éléments
Les pages
Chapitre 2
Plus loin dans la gestion événementielle
Événements personnalisés
Gestion événementielle unique
Déclenchement d'événements
Créer des événements personnalisés
Délégation d'événements

Chapitre 3
Animations et effets
Apparition et disparition
Fondu enchaîné
Aller plus loin
Chapitre 4
Files d'attente et timer
Les files d'attente jQuery
État de la file d'attente
Manipuler la file d'attente
Répéter une animation sans fin
Arrêter et reprendre une animation
Mettre en place un timer
Chapitre 5
Textes et images
Les chaînes de caractères
Les images
Chapitre 6
Formulaires et tableaux
Les formulaires
Les tableaux
Chapitre 7
TP : Mise en forme d'une page Web
Instructions pour réaliser le TP
Correction
Chapitre 8
Un jeu en jQuery
Le document de base
Gérer les déplacements
Détecter les collisions
Ajouter des sons
Le code complet
Chapitre 9
TP : Un jeu de collecte spatiale
Instructions pour réaliser le TP
Correction

jQuery et AJAX

Chapitre 1
Premiers pas avec AJAX
Qu'est-ce qu'AJAX ?
Charger un fichier
Charger une partie d'un fichier
Passer des paramètres à un programme PHP

Requêtes GET et POST
Faire patienter l'utilisateur avec une animation
Chapitre 2
Plus loin avec AJAX
Charger un script et des données JSON
La fonction $.ajax()
Événements associés à une requête AJAX
Chapitre 3
TP : Un chat en jQuery
Instructions pour réaliser le TP
Correction

Les plugins jQuery

Chapitre 1
Trouver et utiliser un plugin
Trouver et utiliser un plugin jQuery
Quelques exemples de plugins
Chapitre 2
jQuery UI
De quoi est capable jQuery UI ?
Déplacer et redimensionner des éléments
Un accordéon
Sélection de date
Des boîtes de dialogue
Afficher des onglets
Animation : une impression de déjà-vu
Animation de couleurs
Modèles de progression
Chapitre 3
Créer un plugin
Le squelette d'un plugin
Un premier plugin
Un plugin plus ambitieux

Annexe

Déboguer le code jQuery
Déboguer avec la fonction alert()
Try et catch
Capturer toutes les erreurs
Déboguer avec Firebug

...AUTRES RESSOURCES...

A connaitre !

- **http://net.tutsplus.com/category/tutorials/javascript-ajax/?tag=tips**
- **http://tympanus.net/codrops/category/tutorials/**
- **http://tutorialzine.com/?s=jquery**
- **http://www.webappers.com/?s=jquery**

A voir !

- http://www.smashingmagazine.com/2011/04/07/useful-javascript-and-jquery-tools-libraries-plugins
- http://webdesigneraid.com/weekly-html5-news-and-inspirations-%E2%80%93-tutorials-tools-resources-and-freebies-v-2/
- http://www.designer-daily.com/15-useful-jquery-plugins-and-tutorials-5207
- http://www.julien-verkest.fr/22/11/2007/240-plugins-jquery
- http://www.hotscripts.com/blog/10-great-html5-experiments-apps/
- http://www.noupe.com/jquery/excellent-jquery-navigation-menu-tutorials.html
- http://www.noupe.com/php/20-useful-php-jquery-tutorials.html
- http://aext.net/2010/04/excellent-jquery-plugins-resources-for-data-presentation-and-grid-layout/
- http://webdesigneraid.com/html5-canvas-graphing-solutions-every-web-developers-must-know/
- http://gestureworks.com/features/open-source-gestures/
- http://edtechdev.wordpress.com/2011/01/14/some-exciting-new-html5javascript-projects/
- http://net.tutsplus.com/articles/web-roundups/30-developers-you-must-subscribe-to-as-a-javascript-junkie/
- http://html5demos.com