Cours de Jquery

Site: Espadon
Cours: JAVASCRIPT
Livre: Cours de Jquery
Imprimé par: Thomas Stegen
Date: mercredi 23 mars 2016, 16:16

Table des matières

1 Introduction au JQuery

jQuery est une librairie JavaScript.

jQuery simplifie grandement la programmation JavaScript.

jQuery est facile à apprendre.

Ce que vous devez savoir

Avant de commencer à étudier Jquery, vous devez avoir des notions avancées de:

- HTML
- CSS
- JavaScript

Copyright

1.1 Préambule

Ajouter jQuery à vos pages Web

il y a plusieurs façon:

- Download la librairie jQuery depuis jQuery.com
- Inclure jQuery à partir d'un CDN, comme Google

Downloader jQuery

il y a 2 versions de jQuery disponibles:

- la version Production - pour vos pages web, version compressée et optimisée
- la version Development - pour tester et devellopper (non compressée et code lisible)

les 2 versions sont téléchargeables depuis jQuery.com.

La librairie jQuery est un simple fichier JavaScript, et vous pouvez le referencer avec la balise HTML <script> dans la balise <head>:

```
<head>
<script src="jquery-1.11.3.min.js"></script>
</head>
```

Conseil: Placer le fichier téléchargé dans le même répertoire que les pages web où vous voulez l'utiliser.

jQuery CDN

Si vous ne voulez pas télécharger et héberger JQuery, vous pouvez l'inclure depuis un CDN (Content Delivery Network).

Google et Microsoft hébergent jQuery.

Google CDN:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
</head>
```

Microsoft CDN:

```
<head>
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.11.3.min.js"></script>
</head>
```

Un gros avantage à utiliser les jQuery de Google ou Microsoft:

Beaucoup d'utilisateurs ont déjà téléchargé le JQuery depuis Google ou Microsoft en visaitant une autre page web. Il sera donc déjà chargé dans le cache, ce qui accélerera le temps de chargement de votre page.

1.2 La syntaxe

En JQuery vous selectionnez (query) des éléments HTML et appliquez des actions dessus.

Syntaxe jQuery

La syntaxe basique est:

$(*selector*).*action*()

- le signe $ pour définir ou accéder à jQuery
- le (*selector*) pour "chercher" l'élément HTML
- l'*action*() pour appliquer l'action sur l'élément(s) HTML

Exemples:

$(this).hide() - cache l'élément courant.

$("p").hide() - cache tous les balises <p>.

$(".test").hide() - cache toutes les balises qui ont la classe "test".

$("#test").hide() - cache toutes les balises ayant l'id="test".

L'évènement Document Ready

En général, on insère les méthodes jQuery dans un évènement document ready:

```
$(document).ready(function(){

  // jQuery methods go here...

});
```

Ceci pour éviter que du code JQuery ne s'execute avant que le document HTML soit entièrement téléchargé.

Il y a certaines actions qui peuvent ne pas marcher si on les execute avant que le document HTML ne soit entièrement chargé:

- Essaye de cacher un élément pas encore crée
- Essayer de récupérer la tailel d'une image qui n'est pas encore chargée

Conseil: une syntaxe encore plus simple pour effectuer la même chose consiste à écrire comme ceci:

```
$(function(){

  // jQuery methods go here...

});
```

Utilisez la syntaxe que vous préférez mais la première est plus facile à comprendre dans votre programme.

1.3 Les sélecteurs

Savoir utiliser les selecteurs est la partie la plus importante de votre apprentissage du jQuery.

Sélecteurs jQuery

ils permettent de sélectionner et manipuler les éléments HTML.

Les sélecteurs jQuery sont utilisés pour "trouver" ou "sélectionner" des éléments HTML en utilisant leur id, classes, types, attributes, valeur des attributes and beaucoup d'autres. Ils sont basés sur des Sélecteurs CSS, en plus de sélecteurs customs.

Tous les sélecteurs en jQuery commencent par le signe dollar et les parenthèses: $().

Les sélecteurs de balise

Vous pouvez sélectionner toutes les balises <p> d'une page HTML comme ceci:

$("p")

Exemple

Lorsqu'un utilisateur clique sur un bouton, toutes les balises <p> seront cachées:

```
$(document).ready(function(){
   $("button").click(function(){
      $("p").hide();
   });
});
```

Le sélecteur #id

Le sélecteur jQuery **#idutilise** l'attribut id d'une balise HTML.

Un **id** est unique sur une page, cela vous permet de trouver un élément unique dans une page HTML.

Pour trouver un élément avec un **id** spécifique, utilisez le # suivi de l'id que vous cherchez:

$("#test")

Exemple

Lorsqu'un utilisateur clique sur un bouton, l'élément avec l'id="test" sera caché:

```
$(document).ready(function(){
   $("button").click(function(){
      $("#test").hide();
   });
});
```

Le sélecteur .class

Pour trouver un élément avec une classe spécifique, il faut utiliser un point suivi par le nom de la classe:

$(".test")

Exemple

Lorsque l'utilisateur clique un bouton, les élements avec la classe class="test" seront cachés:

```
$(document).ready(function(){
   $("button").click(function(){
      $(".test").hide();
   });
});
```

Plus d'exemples de sélecteurs jQuery

Syntax	Description
$("*")	Sélectionne tous les éléments
$(this)	Sélectionne l'élément HTML courant
$("p.intro")	Sélectionne toutes les balises <p> ayant la class="intro"
$("p:first")	Sélectionne la première balise <p>
$("ul li:first")	Sélectionne la première balise de la première balise
$("ul li:first-child")	Sélectionne la première balise de chaque balise
$("[href]")	Sélectionne toutes les balises avec un attribut href
$("a[target='_blank']")	Sélectionne toutes les balises <a> avec un attribut target égal à "_blank"
$("a[target!='_blank']")	Sélectionne toutes les balises <a> avec un attribut target différent de "_blank"
$(":button")	Sélectionne toutes les balises <button> et <input> de type="button"
$("tr:even")	Sélectionne toutes les balises <tr> paires
$("tr:odd")	Sélectionne toutes les balises <tr> impaires

Fonctions JQuery dans un fichier séparé

Vous pouvez écrire vos programmes JQuery dans des fichiers séparés .js

Exemple

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js">
</script>
<script src="my_jquery_functions.js"></script>
</head>
```

1.4 Les évènements

jQuery est spécialement adapté pour utiliser les évènements sur une page HTML.

Que sont les évènements (Events)?

Toutes les actions du visiteur à laquelle la page web peut répondre sont appelées évènements.

Un évènement représente le moment précis où quelque chose arrive.

Exemples:

- bouger la souris au dessus d'un élément
- sélectionner un bouton radio
- cliquer sur un élément

Voici les évènements les plus communs des éléments de la DOM:

Événements de la souris	**Événements du clavier**	**Événements des formulaires**	**Événements du Document/Window**
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

Syntaxe jQuery pour les fonctions liées aux Événements

Pour assigner un évènement Click sur tous les paragraphes d'une page:

$("p").click();

La prochaine étape est de définir ce qui arrive lorsque l'évènement arrive. Vous devez lier une fonction à l'évènement:

```
$("p").click(function(){
  // action goes here!!
});
```

Événements jQuery les plus communs

La méthode $(document).ready()

La méthode **$(document).ready()** permet d'executer une fonction quand le document est intégralement chargé.

La méthode click()

La méthode **click()** permet d'exécuter une fonction lorsque l'utilisateur clique un élément HTML

Par exemple: Lorsqu'un la balise <p> est cliquée, la balise <p> courante est cachée:

Exemple

```
$("p").click(function(){
   $(this).hide();
});
```

La méthode dblclick()

La méthode **dblclick()** permet de lancer une fonction lorsque l'utilisateur double clique sur un élément HTML.

Exemple

```
$("p").dblclick(function(){
   $(this).hide();
});
```

La méthode mouseenter()

La méthode **mouseenter()** est executée lorsque la souris entre sur un élément HTML:

Exemple

```
$("#p1").mouseenter(function(){
   alert("You entered p1!");
});
```

La méthode mouseleave()

La méthode **mouseleave()**est executée lorsque la souris sors d'un élément HTML:

Exemple

```
$("#p1").mouseleave(function(){
   alert("Bye! You now leave p1!");
});
```

La méthode mousedown()

La méthode **mousedown()** est executée lorsque le bouton gauche de la souris est pressé alors que celle ci est sur un élément HTML:

Exemple

```
$("#p1").mousedown(function(){
   alert("Mouse down over p1!");
});
```

La méthode mouseup()

La méthode **mouseup()** est executée lorsque le bouton gauche de la souris est pressé alors que celle ci est sur un élément HTML:

Exemple

```
$("#p1").mouseup(function(){
```

```
    alert("Mouse up over p1!");
});
```

La méthode hover()

La méthode **hover()** est une combinaison des deux méthodes mouseenter() et mouseleave().

La 1ère fonction est executée lorsque la souris entre sur l'élément HTML et la seconde lorsque la souris en sors.

Exemple

```
$("#p1").hover(function(){
    alert("You entered p1!");
},
function(){
    alert("Bye! You now leave p1!");
});
```

La méthode focus()

La méthode **focus()** est executée lorsque un élément de formulaire est mis en focus:

Exemple

```
$("input").focus(function(){
    $(this).css("background-color", "#cccccc");
});
```

La méthode blur()

La méthode **blur()** est executée lorsque un élément de formulaire perd en focus:

Exemple

```
$("input").blur(function(){
    $(this).css("background-color", "#ffffff");
});
```

La méthode on()

La méthode **on()** lie un ou plusieurs évènements pour l'élément sélectionné.

Exemple 1

```
$("p").on("click", function(){
    $(this).hide();
});
```

Exemple 2

```
$("p").on({
    mouseenter: function(){
```

```
        $(this).css("background-color", "lightgray");
    },
    mouseleave: function(){
        $(this).css("background-color", "lightblue");
    },
    click: function(){
        $(this).css("background-color", "yellow");
    }
});
```

2 Les effets Jquery

Tout ce qui concerne les effets avec Jquery

2.1 Hide et Show

Les méthodes hide() et show()

En jQuery, on peut cacher et montrer des éléments HTML avec les méthodes **hide()** et **show()**:

Exemple

```
$("#hide").click(function(){
  $("p").hide();
});

$("#show").click(function(){
  $("p").show();
});
```

Syntax:

$(*selector*).hide(*speed,callback*);

$(*selector*).show(*speed,callback*);

- Le paramètre speed (optionnel) spécifie la vitesse avec laquelle les éléments seront cachés ou montrés, il peut prendre les valeurs suivantes: "**slow**", "**fast**", or le **nombre de millisecondes**.
-
- Le paramètre optionnel **callback** est une fonction qui sera exécutée après que la méthode **hide()** ou **show()** soit finie (voir le chapitre sur les callback).

Exemple

```
$("button").click(function(){
  $("p").hide(1000);
});
```

La méthode toggle()

En jQuery, vous pouvez basculer entre les méthodes **hide()** et **show()** avec la methode **toggle()**.

Les éléments visibles sont cachés et les éléments cachés sont rendus visibles:

Exemple

```
$("button").click(function(){
  $("p").toggle();
});
```

Syntaxe:

$(*selector*).toggle(*speed,callback*);

2.2 Fading

Les méthode Fading

En jQuery il est possible de faire apparaitre ou disparaitre graduellement des éléments.

Les méthodes jQuery sont les suivantes:

- fadeIn()
- fadeOut()
- fadeToggle()
- fadeTo()

La méthode fadeIn()

Pour faire apparaitre graduellement un élément caché.

Syntaxe:

$(*selector*).fadeIn(*speed,callback*);

- Le paramètre optionnel **speed** spécifie la durée de l'effet. Il peut prendre les valeurs "**slow**", "**fast**", ou le **nombre de millisecondes**.
- Le paramètre optionnel **callback** est une fonction qui sera exécutée lorsque l'apparition sera finie.

Exemple

```
$("button").click(function(){
   $("#div1").fadeIn();
   $("#div2").fadeIn("slow");
   $("#div3").fadeIn(3000);
});
```

La méthode fadeOut()

permet de faire disparaitre graduellement un élément visible.

Syntaxe:

$(*selector*).fadeOut(*speed,callback*);

Exemple

```
$("button").click(function(){
   $("#div1").fadeOut();
   $("#div2").fadeOut("slow");
   $("#div3").fadeOut(3000);
});
```

La méthode fadeToggle()

permet de basculer entre les méthodes **fadeIn()** et **fadeOut()**.

Si l'élément est visible, **fadeToggle() l**e fera disparaitre graduellement et inversement.

Syntaxe:

$(*selector*).fadeToggle(*speed,callback*);

Exemple

```
$("button").click(function(){
   $("#div1").fadeToggle();
   $("#div2").fadeToggle("slow");
   $("#div3").fadeToggle(3000);
});
```

La méthode fadeTo()

permet de faire apparaitre ou disparaitre jusqu'à un certain niveau d'opacité (valeur entre 0 et 1).

Syntaxe:

$(*selector*).fadeTo(*speed,opacity,callback*);

- Le paramètre obligatoire opacity spécifie la valeur d'opacité à atteindre (entre 0 et 1).

Exemple

```
$("button").click(function(){
   $("#div1").fadeTo("slow", 0.15);
   $("#div2").fadeTo("slow", 0.4);
   $("#div3").fadeTo("slow", 0.7);
});
```

2.3 Sliding

Les méthodes Sliding

Vous pouvez créer un effet de glissé sur les éléments HTML.

jQuery a les méthodes suivantes:

- slideDown()
- slideUp()
- slideToggle()

La méthode slideDown()

La méthode **slideDown()** est utilisée pour faire glisser un élément vers le bas .

Syntaxe:

$(*selector*).slideDown(*speed,callback*);

Exemple

```
$("#flip").click(function(){
    $("#panel").slideDown();
});
```

La méthode slideUp()

La méthode **slideUp()** est utilisée pour faire glisser un élément vers le haut.

Syntaxe:

$(*selector*).slideUp(*speed,callback*);

Exemple

```
$("#flip").click(function(){
    $("#panel").slideUp();
});
```

La méthode slideToggle()

La méthode **slideToggle()** permet de basculer entre un **slideDown()** et un **slideUp()**.

Si l'élément a été glissé vers la bas il glissera vers le haut et inversement.

Syntaxe:

$(*selector*).slideToggle(*speed,callback*);

Example

```
$("#flip").click(function(){
    $("#panel").slideToggle();
});
```

2.4 Animation

Les animations jQuery - La méthode animate()

Permet de créer des animations personnalisées.

Syntaxe:

$(*selector*).animate({*params*},*speed*,*callback*);

- Le paramètre obligatoire **params** définis les propriétés CSS que l'on souhaite animer.

L'exemple simple suivant permet de déplacer l'élément ** ** vers la droite jusqu'à ce qu'il ai atteint la valeur de l'argument css **left** de 250px:

Exemple

```
$("button").click(function(){
   $("div").animate({left: '250px'});
});
```

Par défaut, tous les éléments HTML ont une position statique (static), et ne peuvent être déplacés.
Pour manipuler leur position, rappelez vous de spécifier, dans un 1er temps, la propriété CSS **position** avec la valeur **relative**, **fixed**, ou **absolute**!

Manipuler de multiples propriétés

de multiples propriétés peuvent être animées en même temps:

Exemple

```
$("button").click(function(){
   $("div").animate({
      left: '250px',
      opacity: '0.5',
      height: '150px',
      width: '150px'
   });
});
```

Est il possible de manipuler TOUTES les propriétés Css avec la méthode animate()?

Oui, presque toutes! Cependant, rappelez vous que tous les noms de propriétés Css doivent être écrits en **lowerCamelCase** lorsque l'on veut les utiliser dans la méthode animate(): Vous devez par exemple écrire **paddingLeft** au lieu de **padding-left**, **marginRight** au lieu de **margin-right**, et ainsi de suite.

https://fr.wikipedia.org/wiki/CamelCase pour la définition de camelCase

De plus, l'animation des couleurs n'est pas inclue dans la librairie jQuery.

Si vous voulez animer les couleurs, il vous faut télécharger le plugin Color Animations plugin depuis jQuery.com.

Utiliser des valeurs relatives

Il est possible de définir des valeurs relatives en mettant += ou -= devant la valeur:

Exemple

```
$("button").click(function(){
  $("div").animate({
    left: '250px',
    height: '+=150px',
    width: '+=150px'
  });
});
```

Utiliser des valeurs prédéfinies

Vous pouvez spécifier la valeur d'une propriété d'animation comme "**show**", "**hide**", ou "**toggle**":

Exemple

```
$("button").click(function(){
  $("div").animate({
    height: 'toggle'
  });
});
```

Utiliser la fonctionnalité Queue

Par défaut, les animations jQuery sont proposées avec une fonctionnalité de queue.

Celà veut dire que si vous écrivez de multiples animations animate() l'une après l'autre, jQuery créera une queue "interne" et exécutera les animations une par une.

Exemple 1

```
$("button").click(function(){
  var div = $("div");
  div.animate({height: '300px', opacity: '0.4'}, "slow");
  div.animate({width: '300px', opacity: '0.8'}, "slow");
  div.animate({height: '100px', opacity: '0.4'}, "slow");
  div.animate({width: '100px', opacity: '0.8'}, "slow");
});
```

Exemple 2

```
$("button").click(function(){
```

```
    var div = $("div");
    div.animate({left: '100px'}, "slow");
    div.animate({fontSize: '3em'}, "slow");
});
```

2.5 Stop

La méthode stop()

permet de stopper une animation ou un effet avant qu'il soit fini.

Cette méthode marche pour toutes les fonctions d'effet JQuery, incluant sliding, fading et les animations.

Syntaxe:

$(*selector*).stop(*stopAll,goToEnd*);

- Le paramètre optionnel **stopAll** spécifie si la queue d'animation doit être vidée ou pas. La valeur par défaut est **false**, ce qui veut dire que seule l'animation en cours d'exécution sera stoppée, permettant aux autres animations de la queue d'être exécutées par la suite.
- Le paramètre optionnel **goToEnd** spécifie si on doit terminer l'animation courant ou pas. La valeur par défaut est **false**.

Exemple

```
$("#stop").click(function(){
   $("#panel").stop();
});
```

2.6 Callback

Une fonction **callback** est exécutée après que l'effet en cours d'exécution soit 100% terminé.

Les fonction Callback

Les instructions JavaScript sont exécutées ligne par ligne. Cependant, avec les effets, l'instruction suivante peut être exécutée alors que l'effet n'est pas encore fini. Cela peut engendrer des erreurs.

Pour éviter cela, vous pouvez créer une fonction **callback**.

Celle ci sera exécutée quand l'effet en cours d'exécution sera fini.

Syntaxe typique:

$(*selector*).hide(*speed,callback*);

Exemples

L'exemple ci-dessous a un paramètre **callback** qui est une fonction qui sera exécutée lorsque l'effet hide() sera fini:

```
$("button").click(function(){
   $("p").hide("slow", function(){
      alert("The paragraph is now hidden");
   });
});
```

L'exemple ci-dessous n'a pas de paramètre **callback**, et la boite d'alerte sera activée avant que l'effet soit fini:

```
$("button").click(function(){
   $("p").hide(1000);
   alert("The paragraph is now hidden");
});
```

2.7 Chaining

En jQuery, il est possible d'enchainer des actions/méthodes.

Cela permet de lancer de multiples méthodes jQuery (sur le même élément) en une seule instruction.

La méthode Chaining

Cette méthode permet au navigateur internet de n'avoir à chercher un élément qu'une seule fois

Pour enchainer les actions, vous devez simplement concaténer les actions aux actions précédentes.

L'exemple ci-dessous en chaine les méthodes css(), slideUp(), et slideDown(). L'élément "p1" change d'abord sa couleur en rouge, puis il glisse vers le haut puis vers les bas:

Exemple

```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

Il est possible d'enchainer autant de méthodes que désiré.

Tip: Lorsque l'on enchaine des méthodes, les lignes de codes peuvent devenir longues. Cependant, jQuery n'est pas très strict sur la syntaxe, vous pouvez la formater comme vous le désirez , en utilisant des sauts à la ligne ou de indentations pour rendre le code plus lisible.

Exemple

```
$("#p1").css("color", "red")
  .slideUp(2000)
  .slideDown(2000);
```

3 JQuery et HTML

Comment manipuler les objets HTML avec JQuery

3.1 Get Content and Attributes

jQuery permet de changer et manipuler les balises HTML et leurs attributs.

Manipulation de la DOM (Document Object Model)

Manipuler la DOM est l'une des fonctionnalités les plus importantes du jQuery.

DOM = Document Object Model

La DOM définie un standard pour accéder aux documents HTML et XML:

"Le W3C Document Object Model (DOM) est une plateforme et une interface "language-neutral" qui permet aux programmes et aux scripts d'accéder dynamiquement et de mettre à jour le contenu, la structure et le style d'un document."

Get Content - text(), html(), and val()

Les trois méthodes jQuery pour manipuler la DOM sont:

- text() - Met en place ou renvoi le contenu texte d'une balise
- html() - Met en place ou renvoi le contenu d'une balise (y compris la balise HTML elle même)
- val() - Met en place ou renvoi la valeur d'un champ

Exemple

```
$("#btn1").click(function(){
  alert("Text: " + $("#test").text());
});
$("#btn2").click(function(){
  alert("HTML: " + $("#test").html());
});
```

Exemple

```
$("#btn1").click(function(){
  alert("Value: " + $("#test").val());
});
```

Get Attributes - attr()

La méthode attr() est utilisée pour récupérer des valeurs d'attributs.

L'exemple suivant montre comment récupérer la valeur d'un attribut href d'un lien:

Exemple

```
$("button").click(function(){
  alert($("#w3s").attr("href"));
});
```

3.2 Set Content and Attributes

Set Content - text(), html(), et val()

Nous utiliserons les mêmes trois méthodes que le chapitre précédent pour changer un contenu:

- text()
- html()
- val()

Exemple

```
$("#btn1").click(function(){
  $("#test1").text("Hello world!");
});
$("#btn2").click(function(){
  $("#test2").html("<b>Hello world!</b>");
});
$("#btn3").click(function(){
  $("#test3").val("Dolly Duck");
});
```

Utiliser une fonction Callback avec text(), html(), et val()

Les 3 méthodes ci-dessus: text(), html(), et val(), peuvent aussi être utilisées avec une fonction callback. La fonction callback as deux paramètres: L'indice de l'élément courant dans la liste des éléments sélectionnés et la valeur originale. Vous pouvez alors renvoyer la chaine de caractère que vous souhaitez utiliser comme nouvelle valeur de la fonction.

Exemple

```
$("#btn1").click(function(){
  $("#test1").text(function(i, origText){
    return "Old text: " + origText + " New text: Hello world!
    (index: " + i + ")";
  });
});

$("#btn2").click(function(){
  $("#test2").html(function(i, origText){
    return "Old html: " + origText + " New html: Hello <b>world!</b>
    (index: " + i + ")";
  });
});
```

Donner une valeur à un attribut - attr()

La méthode attr() permet de donner une valeur (changer) à un attribut.

Exemple

```
$("button").click(function(){
   $("#w3s").attr("href", "http://www.w3schools.com/jquery");
});
```

La méthode attr() permet de donner une valeur à plusieurs attributs en même temps.

Exemple

```
$("button").click(function(){
   $("#w3s").attr({
      "href" : "http://www.w3schools.com/jquery",
      "title" : "W3Schools jQuery Tutorial"
   });
});
```

Fonction Callback et attr()

La méthode attr() peut aussi être utilisée avec une fonction callback. La fonction callback as deux paramètres: L'indice de l'élément courant dans la liste des éléments sélectionnés et la valeur originale. Vous pouvez alors renvoyer la chaine de caractère que vous souhaitez utiliser comme nouvelle valeur de la fonction.

Exemple

```
$("button").click(function(){
   $("#w3s").attr("href", function(i, origValue){
      return origValue + "/jquery";
   });
});
```

3.3 Add Elements

En jQuery, il est très facile de rajouter de nouvelles balises/contenu.

Ajouter de nouveaux contenus HTML

Nous allons jeter un œil à quatre méthodes jQuery pour cela:

- append() - Insert du contenu à la fin de l'élément sélectionné
- prepend() - Insert du contenu au début de l'élément sélectionné
- after() - Insert du contenu après l'élément sélectionné
- before() - Insert du contenu avant l'élément sélectionné

La méthode append()

Insère du contenu A LA FIN de l'élément HTML sélectionné.

Exemple

```
$("p").append("Some appended text.");
```

La méthode prepend()

Insère du contenu AU DEBUT de l'élément HTML sélectionné.

Exemple

```
$("p").prepend("Some prepended text.");
```

Ajouter plusieurs nouveaux éléments avec append() et prepend()

Dans les deux exemples ci dessus, nous avons inséré du text/HTML au début/fin de l'élément HTML sélectionné.

Cependant, les deux méthodes append() et prepend() peuvent avoir une infinité de nouveaux éléments comme paramètres.
Les nouveaux éléments peuvent être générés avec du text/HTML comme précédemment, mais aussi avec JQuery, du code javascript et des éléments de la DOM.
Dans l'exemple suivant, nous créons de nouveaux éléments avant de les intégrer avec la méthode append():

Exemple

```
function appendText() {
  var txt1 = "<p>Text.</p>";              // Crée un élément avec HTML
  var txt2 = $("<p></p>").text("Text.");   // avec JQuery
  var txt3 = document.createElement("p");  // avec la DOM
  txt3.innerHTML = "Text.";
  $("p").append(txt1, txt2, txt3);         // Append les nouveaux éléments
```

```
}
```

Les méthodes after() et before()

La méthode after() insert du contenu APRES l'élément HTML sélectionné.

La méthode before() insert du contenu AVANT l'élément HTML sélectionné.

Exemple

```
$("img").after("Some text after");

$("img").before("Some text before");
```

Ajouter plusieurs nouveaux éléments avec after() et before()

Comme précédemment :

Exemple

```
function afterText() {
   var txt1 = "<b>I </b>";                  // Crée un élément avec HTML
   var txt2 = $("<i></i>").text("love ");     // avec JQuery
   var txt3 = document.createElement("b");    // avec la DOM DOM
   txt3.innerHTML = "jQuery!";
   $("img").after(txt1, txt2, txt3);          // Insert les nouveaux éléments aprèsr <img>
}
```

3.4 Remove Elements

En jQuery, il est facile d'enlever des éléments HTML existant.

Enlever des balises/contenu

Pour enlever des balises et du contenu il y a deux méthodes jQuery:

- remove() - Enlève la balise sélectionnée (et ses enfants)
- empty() - Enlève les enfants de la balise sélectionnée

La méthode remove()

Example

$("#div1").remove();

La méthode empty()

Elle enlève les enfants de la balise sélectionnée.

Exemple

$("#div1").empty();

Filtrer les balises à enlever

La méthode **remove()** accepte aussi un paramètre, qui permet de filtrer les éléments à enlever.

Ce paramètre peut être n'importe quel sélecteur Jquery.

L'exemple suivant enlève toutes les balises <p> avec la class="test":

Exemple

$("p").remove(".test");

L'exemple suivant enlève toutes les balises <p> avec la class="test" et la class="demo":

Exemple

$("p").remove(".test, .demo");

3.5 Get and Set CSS Classes

En jQuery, il est facile de manipuler les éléments Css.

Manipuler les CSS

jQuery as plusieurs méthodes pour la manipulation de CSS:

- addClass() - Ajoute une ou plusieurs classes à la balise sélectionnée
- removeClass() - Enlève une ou plusieurs classes à la balise sélectionnée
- toggleClass() - bascule entre ajouter/enlever des classes à la balise sélectionnée
- css() - donne une valeur ou renvoi l'attribut de style Css

Exemple de feuille de style

La feuille de style suivante sera utilisée pour tous les exemples de ce chapitre:

```
.important {
   font-weight: bold;
   font-size: xx-large;
}

.blue {
   color: blue;
}
```

La méthode addClass()

L'exemple suivant montre comment ajouter des attributs de classe à différentes balises:

Exemple

```
$("button").click(function(){
   $("h1, h2, p").addClass("blue");
   $("div").addClass("important");
});
```

Vosu pouvez aussi spécifier plusieurs classes avec la méthode addClass():

Exemple

```
$("button").click(function(){
   $("#div1").addClass("important blue");
});
```

La méthode removeClass()

Exemple

```
$("button").click(function(){
    $("h1, h2, p").removeClass("blue");
});
```

La méthode toggleClass()

Exemple

```
$("button").click(function(){
    $("h1, h2, p").toggleClass("blue");
});
```

3.6 css() Method

La méthode css()

La méthode css() met une valeur ou renvoi une ou plusieurs propriétés de style de la balise sélectionnée.

Renvoyer une propriété CSS

Pour cela il faut utiliser la syntaxe suivante:

css("*propertyname*");

L'exemple suivant renvoi la valeur de background-color de la première balise <p> sélectionné:

Exemple

$("p").css("background-color");

Donner une valeur à une propriété CSS

La syntaxe est la suivante:

css("*propertyname*","*value*");

L'exemple suivant fixera la valeur du background-color pour TOUT les éléments sélectionnés:

Exemple

$("p").css("background-color", "yellow");

Donner une valeur à plusieurs propriétés CSS

utilisez la syntaxe suivante:

css({"*propertyname*":"*value*","*propertyname*":"*value*",...});

Exemple

$("p").css({"background-color": "yellow", "font-size": "200%"});

3.7 Dimensions

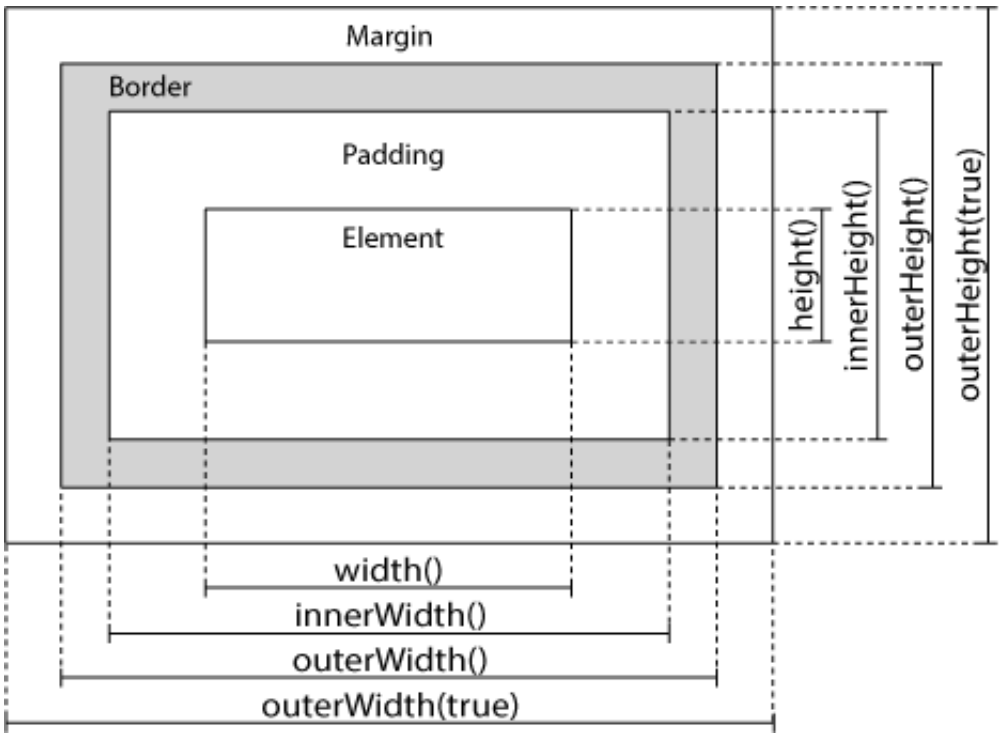
En jQuery, il est facile de travailler avec les dimensions des éléments et des fenêtre du navigateur.

Les méthodes de Dimension

Les méthodes suivantes permettent de travailler sur les dimensions:

- width()
- height()
- innerWidth()
- innerHeight()
- outerWidth()
- outerHeight()

Les dimensions



Les méthodes width() et height()

La méthode width() fixe ou renvoi la largeur d'un élément (à l'exclusion des padding, border et margin).

La méthode height() fixe ou renvoi la hauteur d'un élément (à l'exclusion des padding, border et margin).

L'exemple suivant renvoi la largeur et hauteur d'un :

Exemple

```
$("button").click(function(){
  var txt = "";
  txt += "Width: " + $("#div1").width() + "</br>";
```

```
    txt += "Height: " + $("#div1").height();
    $("#div1").html(txt);
});
```

---

jQuery innerWidth() and innerHeight() Methods

La méthode innerWidth() renvoi la largeur d'un élément (incluant le padding).

La méthode innerHeight() renvoi la hauteur d'un élément (incluant le padding).

Exemple

```
$("button").click(function(){
    var txt = "";
    txt += "Inner width: " + $("#div1").innerWidth() + "</br>";
    txt += "Inner height: " + $("#div1").innerHeight();
    $("#div1").html(txt);
});
```

Les méthodes outerWidth() et outerHeight()

La méthode outerWidth() renvoi la largeur d'un élément (incluant le padding et le border).

La méthode outerHeight() renvoi la hauteur d'un élément (incluant le padding et le border).

Exemple

```
$("button").click(function(){
    var txt = "";
    txt += "Outer width: " + $("#div1").outerWidth() + "</br>";
    txt += "Outer height: " + $("#div1").outerHeight();
    $("#div1").html(txt);
});
```

La méthode outerWidth(true) renvoi la largeur d'un élément (incluant le padding, le border, et le margin).

La méthode outerHeight(true) renvoi la hauteur d'un élément (incluant le padding, le border et le margin).

Exemple

```
$("button").click(function(){
    var txt = "";
    txt += "Outer width (+margin): " + $("#div1").outerWidth(true) + "</br>";
    txt += "Outer height (+margin): " + $("#div1").outerHeight(true);
    $("#div1").html(txt);
});
```

Plus de width() et height()

L'exemple suivant renvoi la largeur et hauteur du document (le document HTML) et de la fenêtre (du navigateur):

Exemple

```
$("button").click(function(){
   var txt = "";
   txt += "Document width/height: " + $(document).width();
   txt += "x" + $(document).height() + "\n";
   txt += "Window width/height: " + $(window).width();
   txt += "x" + $(window).height();
   alert(txt);
});
```

L'exemple suivant fixe la largeur et la hauteur d'une balise :

Exemple

```
$("button").click(function(){
   $("#div1").width(500).height(500);
});
```

4 Références

Ce chapitre contient tous les éléments JQuery dont vous pouvez avoir besoin

4.1 Les Sélecteurs

Sélecteurs jQuery

(Liens W3schools.com)

Sélecteur	Exemple	Sélectionne
*	$("*")	Toutes les balises
#id	$("#lastname")	Les balises avec id="lastname"
.class	$(".intro")	Toutes les balises avec class="intro"
.class,.class	$(".intro,.demo")	Toutes les balises avec la classe "intro" or "demo"
element	$("p")	Toutes les balises **<p>**
el1,*el2*,*el3*	$("h1,div,p")	Toutes les balises **<h1>, et <p>**
:first	$("p:first")	La première balise **<p>**
:last	$("p:last")	La dernière balise **<p>**
:even	$("tr:even")	Toutes les balises **<tr>** paires
:odd	$("tr:odd")	Toutes les balises **<tr>** impaires
:first-child	$("p:first-child")	Toutes les balises **<p>** qui sont le premier enfant de son parent
:first-of-type	$("p:first-of-type")	Toutes les balises **<p>** qui sont la première balise <p> de son parent
:last-child	$("p:last-child")	Toutes les balises **<p>** qui sont le dernier enfant de son parent
:last-of-type	$("p:last-of-type")	Toutes les balises **<p>** qui sont la dernière balise <p> de son parent
:nth-child(*n*)	$("p:nth-child(2)")	Toutes les balises **<p>** qui sont le 2nd enfant de son parent
:nth-last-child(*n*)	$("p:nth-last-child(2)")	Toutes les balises **<p>** qui sont le 2nd enfant de son parent en comptant depuis le dernier enfant
:nth-of-type(*n*)	$("p:nth-of-type(2)")	Toutes les balises **<p>** qui sont la 2nd balise **<p>** de son parent
:nth-last-of-type(*n*)	$("p:nth-last-of-type(2)")	Toutes les balises **<p>** qui sont la 2nd balise **<p>** de son parent, en partant du dernier enfant
:only-child	$("p:only-child")	Toutes les balises **<p>** qui sont le seul enfant de leur parent
:only-of-type	$("p:only-of-type")	Toutes les balises **<p>** qui sont le seul enfant de son type de leur parent
parent > child	$("div > p")	Toutes les balises **<p>** qui sont un enfant direct de la balise ** **
parent descendant	$("div p")	Toutes les balises **<p>** qui sont un descendants de la balise ** **
element + next	$("div + p")	Toutes les balises **<p>** qui sont juste après chaque balise ** **

element ~ siblings	$("div ~ p")	Toutes les balises **<p>** qui sont frère d'une balise ** **
:eq(*index*)	$("ul li:eq(3)")	La 4ème balise **li** d'une liste
:gt(*no*)	$("ul li:gt(3)")	Les balises **li** d'indice supérieur ou égal à 3
:lt(*no*)	$("ul li:lt(3)")	Les balises **li** d'indice inférieur à 3
:not(*selector*)	$("input:not(:empty)")	Toutes les balises **input** non vides
:header	$(":header")	Toutes les balises de titre **<h1>, <h2> ...**
:animated	$(":animated")	Toutes les balises animées
:focus	$(":focus")	Les balises qui sont sous le focus
:contains(*text*)	$(":contains('Hello')")	Toutes les balises qui contiennent le texte "Hello"
:has(*selector*)	$("div:has(p)")	Toutes les balises ** ** qui ont une balise **<p>**
:empty	$(":empty")	Toutes les balises vides
:parent	$(":parent")	Toutes les balises qui sont parent d'une autre balise
:hidden	$("p:hidden")	Toutes les balises **<p>** cachées
:visible	$("table:visible")	Tous les tableaux visibles
:root	$(":root")	La balise **root** du document
:lang(*language*)	$("p:lang(de)")	Toutes les balises **<p>** avec un attribut **lang** commençant par "de"
[*attribute*]	$("[href]")	Toutes les balises avec un attribut **href**
[*attribute*=*value*]	$("[href='default.htm']")	Toutes les balises avec un attribut **href** égal à "default.htm"
[*attribute*!=*value*]	$("[href!='default.htm']")	Toutes les balises avec un attribut **href** différent de "default.htm"
[*attribute*$=*value*]	$("[href$='.jpg']")	Toutes les balises avec un attribut **href** se terminant par ".jpg"
[*attribute*|=*value*]	$("[title|='Tomorrow']")	Toutes les balises avec un attribut **title** egal à 'Tomorrow', ou commençant par 'Tomorrow' suivit de]
[*attribute*^=*value*]	$("[title^='Tom']")	Toutes les balises avec un attribut **title** commençant par "Tom"
[*attribute*~=*value*]	$("[title~='hello']")	Toutes les balises avec un attribut **title** qui contient le mot spécifique "hello"
[*attribute**=*value*]	$("[title*='hello']")	Toutes les balises avec un attribut **title** qui contient le mot "hello"
:input	$(":input")	Toutes les balises **input**
:text	$(":text")	Toutes les balises **input** avec **type="text"**
:password	$(":password")	Toutes les balises **input** avec **type="password"**
:radio	$(":radio")	Toutes les balises **input** avec **type="radio"**
:checkbox	$(":checkbox")	Toutes les balises **input** avec **type="checkbox"**
:submit	$(":submit")	Toutes les balises **input** avec **type="submit"**
:reset	$(":reset")	Toutes les balises **input** avec **type="reset"**
:button	$(":button")	Toutes les balises **input** avec **type="button"**
:image	$(":image")	Toutes les balises **input** avec **type="image"**

:file	$(":file")	Toutes les balises **input** avec **type="file"**
:enabled	$(":enabled")	Toutes les balises **input enabled**
:disabled	$(":disabled")	Toutes les balises **input disabled**
:selected	$(":selected")	Toutes les balises **input selected**
:checked	$(":checked")	Toutes les balises **input checked**

4.2 Les évènements

Les méthodes pour les évènements (Events)

Les méthodes d'évènements se déclenchent ou attachent une fonction à un évènement de l'élément sélectionné.

La table suivante liste toutes les méthodes jQuery concernant les évènements. Suivez le lien pour en savoir plus, malheureusement je n'ai pas la patience de tout traduire pour vous...

(Liens W3schools.com)

Method / Property	Description
bind()	Attache un évènement à un élément
blur()	Attache/Déclenche l'évènement blur
change()	Attache/Déclenche l'évènement change
click()	Attache/Déclenche l'évènement click
dblclick()	Attache/Déclenche l'évènement double click
delegate()	Attaches a handler to current, or future, specified child elements of the matching elements
die()	Retiré de la version 1.9.
error()	Dépréciée dans la version 1.8.
event.currentTarget	The current DOM element within the event bubbling phase
event.data	Contains the optional data passed to an event method when the current executing handler is bound
event.delegateTarget	Returns the element where the currently-called jQuery event handler was attached
event.isDefaultPrevented()	Returns whether event.preventDefault() was called for the event object
event.isImmediatePropagationStopped()	Returns whether event.stopImmediatePropagation() was called for the event object
event.isPropagationStopped()	Returns whether event.stopPropagation() was called for the event object
event.namespace	Returns the namespace specified when the event was triggered
event.pageX	Returns the mouse position relative to the left edge of the document
event.pageY	Returns the mouse position relative to the top edge of the document
event.preventDefault()	Prevents the default action of the event
event.relatedTarget	Returns which element being entered or exited on mouse movement.
event.result	Contains the last/previous value returned by an event handler triggered by the specified event
event.stopImmediatePropagation()	Prevents other event handlers from being called
event.stopPropagation()	Prevents the event from bubbling up the DOM tree, preventing any parent handlers from being notified of the event
event.target	Returns which DOM element triggered the event

event.timeStamp	Returns the number of milliseconds since January 1, 1970, when the event is triggered
event.type	Returns which event type was triggered
event.which	Returns which keyboard key or mouse button was pressed for the event
focus()	Attache/Déclenche l'évènement focus
focusin()	Attaches an event handler to the focusin event
focusout()	Attaches an event handler to the focusout event
hover()	Attaches two event handlers to the hover event
keydown()	Attache/Déclenche l'évènement keydown
keypress()	Attache/Déclenche l'évènement keypress
keyup()	Attache/Déclenche l'évènement keyup
live()	Removed in version 1.9. Adds one or more event handlers to current, or future, selected elements
load()	Deprecated in version 1.8. Attaches an event handler to the load event
mousedown()	Attache/Déclenche l'évènement mousedown
mouseenter()	Attache/Déclenche l'évènement mouseenter
mouseleave()	Attache/Déclenche l'évènement mouseleave
mousemove()	Attache/Déclenche l'évènement mousemove
mouseout()	Attache/Déclenche l'évènement mouseout
mouseover()	Attache/Déclenche l'évènement mouseover
mouseup()	Attache/Déclenche l'évènement mouseup
off()	Removes event handlers attached with the on() method
on()	Attaches event handlers to elements
one()	Adds one or more event handlers to selected elements. This handler can only be triggered once per element
$.proxy()	Takes an existing function and returns a new one with a particular context
ready()	Specifies a function to execute when the DOM is fully loaded
resize()	Attache/Déclenche l'évènement resize
scroll()	Attache/Déclenche l'évènement scroll
select()	Attache/Déclenche l'évènement select
submit()	Attache/Déclenche l'évènement submit
toggle()	Removed in version 1.9. Attaches two or more functions to toggle between for the click event
trigger()	Triggers all events bound to the selected elements
triggerHandler()	Triggers all functions bound to a specified event for the selected elements
unbind()	Removes an added event handler from selected elements
undelegate()	Removes an event handler to selected elements, now or in the future
unload()	Deprecated in version 1.8. Attaches an event handler to the unload event

4.3 Les effets

Les méthodes pour les effets JQuery

La table suivante liste les méthodes pour créer des effets JQuery.

(Liens W3schools.com)

Method	Description
animate()	Execute une animation sur l'élément sélectionné
clearQueue()	Retire toutes les fonctions restant dans la queue pour l'élément sélectionné
delay()	Fixe un délai pour toutes les fonction dans la queue pour l'élément sélectionné
dequeue()	Retire la prochaine fonction dans la queue et exécute la fonction
fadeIn()	Fait apparaitre progressivement l'élément sélectionné
fadeOut()	Fait disparaitre progressivement l'élément sélectionné
fadeTo()	Fait apparaitre (ou disparaitre) progressivement vers une opacité donnée
fadeToggle()	Bascule entre le fadeIn() et le fadeOut()
finish()	Stoppe, enlève et fini toutes les queues d'animations de l'élément sélectionné
hide()	Cache l'élément sélectionné
queue()	Montre les fonction dans la queue de l'élément sélectionné
show()	Montre l'élément sélectionné
slideDown()	Slides-down (montre en descendant) l'élément sélectionné
slideToggle()	Bascule entre le slideUp() et le slideDown()
slideUp()	Slides-up (cache en montant) l'élément sélectionné
stop()	Stoppe l'animation en train d'être exécutée pour l'élément sélectionné
toggle()	Bascule entre le hide() et le show()

4.4 HTML / CSS

Les méthodes HTML / CSS

The following table lists all the methods used to manipulate the HTML and CSS.

The methods below work for both HTML and XML documents. Exception: the html() method.

(Liens W3schools.com)

Method	Description
addClass()	Adds one or more class names to selected elements
after()	Inserts content after selected elements
append()	Inserts content at the end of selected elements
appendTo()	Inserts HTML elements at the end of selected elements
attr()	Sets or returns attributes/values of selected elements
before()	Inserts content before selected elements
clone()	Makes a copy of selected elements
css()	Sets or returns one or more style properties for selected elements
detach()	Removes selected elements (keeps data and events)
empty()	Removes all child nodes and content from selected elements
hasClass()	Checks if any of the selected elements have a specified class name
height()	Sets or returns the height of selected elements
html()	Sets or returns the content of selected elements
innerHeight()	Returns the height of an element (includes padding, but not border)
innerWidth()	Returns the width of an element (includes padding, but not border)
insertAfter()	Inserts HTML elements after selected elements
insertBefore()	Inserts HTML elements before selected elements
offset()	Sets or returns the offset coordinates for selected elements (relative to the document)
offsetParent()	Returns the first positioned parent element
outerHeight()	Returns the height of an element (includes padding and border)
outerWidth()	Returns the width of an element (includes padding and border)
position()	Returns the position (relative to the parent element) of an element
prepend()	Inserts content at the beginning of selected elements
prependTo()	Inserts HTML elements at the beginning of selected elements
prop()	Sets or returns properties/values of selected elements
remove()	Removes the selected elements (including data and events)
removeAttr()	Removes one or more attributes from selected elements
removeClass()	Removes one or more classes from selected elements
removeProp()	Removes a property set by the prop() method
replaceAll()	Replaces selected elements with new HTML elements
replaceWith()	Replaces selected elements with new content
scrollLeft()	Sets or returns the horizontal scrollbar position of selected elements
scrollTop()	Sets or returns the vertical scrollbar position of selected elements
text()	Sets or returns the text content of selected elements
toggleClass()	Toggles between adding/removing one or more classes from selected elements
unwrap()	Removes the parent element of the selected elements

val()	Sets or returns the value attribute of the selected elements (for form elements)
width()	Sets or returns the width of selected elements
wrap()	Wraps HTML element(s) around each selected element
wrapAll()	Wraps HTML element(s) around all selected elements
wrapInner()	Wraps HTML element(s) around the content of each selected element

4.5 Traversing Methods

jQuery Traversing Methods

(Liens W3schools.com)

Method	Description
add()	Adds elements to the set of matched elements
addBack()	Adds the previous set of elements to the current set
andSelf()	Deprecated in version 1.8. An alias for addBack()
children()	Returns all direct children of the selected element
closest()	Returns the first ancestor of the selected element
contents()	Returns all direct children of the selected element (including text and comment nodes)
each()	Executes a function for each matched element
end()	Ends the most recent filtering operation in the current chain, and return the set of matched elements to its previous state
eq()	Returns an element with a specific index number of the selected elements
filter()	Reduce the set of matched elements to those that match the selector or pass the function's test
find()	Returns descendant elements of the selected element
first()	Returns the first element of the selected elements
has()	Returns all elements that have one or more elements inside of them
is()	Checks the set of matched elements against a selector/element/jQuery object, and return true if at least one of these elements matches the given arguments
last()	Returns the last element of the selected elements
map()	Passes each element in the matched set through a function, producing a new jQuery object containing the return values
next()	Returns the next sibling element of the selected element
nextAll()	Returns all next sibling elements of the selected element
nextUntil()	Returns all next sibling elements between two given arguments
not()	Remove elements from the set of matched elements
offsetParent()	Returns the first positioned parent element
parent()	Returns the direct parent element of the selected element
parents()	Returns all ancestor elements of the selected element
parentsUntil()	Returns all ancestor elements between two given arguments
prev()	Returns the previous sibling element of the selected element
prevAll()	Returns all previous sibling elements of the selected element
prevUntil()	Returns all previous sibling elements between two given arguments
siblings()	Returns all sibling elements of the selected element
slice()	Reduces the set of matched elements to a subset specified by a range of indices

4.6 Méthodes diverses

jQuery Misc Methods

(Liens W3schools.com)

Method	Description
data()	Attaches data to, or gets data from, selected elements
each()	Execute a function for each matched element
get()	Get the DOM elements matched by the selector
index()	Search for a given element from among the matched elements
$.noConflict()	Release jQuery's control of the $ variable
$.param()	Create a serialized representation of an array or object (can be used as URL query string for AJAX requests)
removeData()	Removes a previously-stored piece of data
size()	Deprecated in version 1.8. Return the number of DOM elements matched by the jQuery selector
toArray()	Retrieve all the DOM elements contained in the jQuery set, as an array