jQuery :

Table des matières

[Rappels CSS 3](#_Toc466999020)

[Quelle est la nature de l'objet retourné ? 3](#_Toc466999021)

[Plus loin dans la sélection d’éléments : 4](#_Toc466999022)

[Sélecteurs d’attributs 4](#_Toc466999023)

[Sélecteurs hiérarchiques 4](#_Toc466999024)

[Pseudo-sélecteurs d’éléments sélectionnés 5](#_Toc466999025)

[Sélecteurs d’éléments particuliers 5](#_Toc466999026)

[Pseudo-sélecteurs d’éléments liés aux formulaires : 5](#_Toc466999027)

[Pseudo-sélecteurs d’éléments liés aux tableaux 6](#_Toc466999028)

[Modifier contenu d’un élément 7](#_Toc466999029)

[Getter et Setter : 7](#_Toc466999030)

[Modifier les attributs : 7](#_Toc466999031)

[Lire ou ajouter : 7](#_Toc466999032)

[Supprimer : 8](#_Toc466999033)

[Travailler avec l’attribut class : 8](#_Toc466999034)

[Test de l’existence d’une classe : 8](#_Toc466999035)

[Travailler avec les formulaires : 9](#_Toc466999036)

[Travailler avec des valeurs stockées : 9](#_Toc466999037)

[Position et taille des éléments : 10](#_Toc466999038)

[Modifier la position : 10](#_Toc466999039)

[Modifier la taille : 11](#_Toc466999040)

[Connaitre les dimensions : 12](#_Toc466999041)

[Associer des données aux balises : 12](#_Toc466999042)

[Insérer et remplacer des éléments du DOM 13](#_Toc466999043)

[Insérer du contenu 13](#_Toc466999044)

[Remplacer du contenu : 13](#_Toc466999045)

[Insérer des éléments : 13](#_Toc466999046)

[Déplacer du contenu : 14](#_Toc466999047)

[Dupliquer des éléments : 14](#_Toc466999048)

[Entourer des éléments : 15](#_Toc466999049)

[Supprimer des éléments : 15](#_Toc466999050)

# Rappels CSS

| **Sélecteur CSS** | **Sélecteur jQuery** | **Signification** |
| --- | --- | --- |
| ul | $('ul') | Les balises <ul> |
| ul.bleu | $('ul.bleu') | La balise <ul> de classe bleu |
| div ul | $('div ul') | La balise <ul> contenue dans la balise <div> |
| div ul li[class="pair"] | $('div ul li[class="pair"]') | La balise <li> contenue dans une balise <ul>, elle-même contenue dans une balise <div>, et dont l'attribut class vaut pair |
| li[class] | $('li[class]') | Les balises <li> qui possèdent un attribut class |
| li[class="impair"] | $('li[class="impair"]') | Les balises <li> qui possèdent un attribut classde valeur impair |
| \* | $('\*') | Toutes les balises du document |

### Quelle est la nature de l'objet retourné ?

Le résultat retourné par la fonction $() est un objet jQuery. Cet objet ressemble à un tableau : il a une propriété length et les éléments sélectionnés peuvent être accédés par un indice. Par exemple :

* $('a').length retourne le nombre de liens hypertextes contenus dans la page.
* $('ul.bleu').length retourne le nombre de balises <ul> de classe bleu.
* $('li[class="impair"]').length retourne le nombre de balises <li> qui ont un attribut class de valeur impair.
* $('body').length retourne « 1 » car le document contient une seule balise <body>.

Pour accéder à un des éléments sélectionnés, précisez son indice entre crochets à la suite du sélecteur. Par exemple :

* $('a')[0] retourne le premier lien hypertexte de la page.
* $('ul.bleu')[3] retourne la quatrième balise <ul> de classe bleu.
* $('body')[0] est équivalent à document.body.

# Plus loin dans la sélection d’éléments :

### Sélecteurs d’attributs

| **Sélecteur** | **Éléments sélectionnés** |
| --- | --- |
| ['nom\*="valeur"'] | Éléments qui possèdent un attribut nom qui contient (partiellement ou totalement) la valeur spécifiée. |
| ['nom~="valeur"'] | Éléments qui possèdent un attribut nom qui contient la valeur spécifiée, délimité par des espaces. |
| ['nom$="valeur"'] | Éléments qui possèdent un attribut nom qui se termine par la valeur spécifiée. |
| ['nom!="valeur"'] | Éléments qui ne possèdent pas l'attribut nom, ou qui possèdent un attribut nomdifférent de la valeur spécifiée. |
| ['nom^="valeur"'] | Éléments qui possèdent un attribut nom qui commence par la valeur spécifiée. |

### Sélecteurs hiérarchiques

| **Sélecteur** | **Éléments sélectionnés** |
| --- | --- |
| ('p > e') | Éléments e directement descendants d'éléments p |
| ('p + e') | Éléments e directement précédés d'un élément p |
| ('p ~ e') | Éléments e précédés d'un élément p |
| :empty | Éléments qui n'ont pas d'enfant |
| :first-child | Premier enfant |
| :first | Premier élément |
| :last-child | Dernier enfant |
| :last | Le dernier élément de la sélection |
| :nth-child() | Élément qui est l'énième enfant de son parent |
| :only-child | Éléments qui sont enfants uniques de leur parent |

### Pseudo-sélecteurs d’éléments sélectionnés

| **Sélecteur** | **Éléments sélectionnés** |
| --- | --- |
| :even | Éléments pairs |
| :odd | Éléments impairs |
| :eq() | Élément dont l'index est spécifié |
| :gt() | Éléments dont l'index est supérieur à (*greater than*) l'index spécifié |
| :lt() | Éléments dont l'index est inférieur à (*lower than*) l'index spécifié |

### Sélecteurs d’éléments particuliers

| **Sélecteur** | **Éléments sélectionnés** |
| --- | --- |
| :header | Tous les titres <h1> à <h6> |
| :hidden | Éléments cachés |
| :visible | Éléments visibles |
| :not() | Éléments qui ne correspondent pas au sélecteur spécifié |

### Pseudo-sélecteurs d’éléments liés aux formulaires :

| **Pseudo-sélecteur** | **Éléments sélectionnés** |
| --- | --- |
| :input | Tous les éléments de type input, textarea, select et button |
| :button | Éléments de type button |
| :checkbox | Éléments de type checkbox |
| :checked | Éléments qui sont cochés |
| :radio | Éléments de type radio |
| :reset | Éléments de type reset |
| :image | Tous les boutons de type image |
| :submit | Éléments de type submit |
| :text | Éléments de type text |
| :password | Éléments de type password |
| :selected | Éléments sélectionnés |
| :focus | Sélectionne l'élément s'il a le focus |
| :enabled | Éléments validés |

### Pseudo-sélecteurs d’éléments liés aux tableaux

| **Pseudo-sélecteur** | **Éléments sélectionnés** |
| --- | --- |
| :first | Premier élément |
| :last | Dernier élément |
| :eq() | Élément dont l'index est spécifié |
| :gt() | Éléments dont l'index est supérieur à l'index spécifié |
| :lt() | Éléments dont l'index est inférieur à l'index spécifié |
| :even | Éléments d'index pair |
| :odd | Éléments d'index impair |
| :empty | Éléments qui n'ont pas d'enfant |
| :not() | Éléments qui ne correspondent pas au sélecteur spécifié |

# Modifier contenu d’un élément

## Getter et Setter :

$('h2').css('font-size'); Getter

$('h2').css('font-size','2em'); Setter

Exemple :

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Sélecteurs CSS</title>

</head>

<body>

<div id="listes">

<a href="http://api.jquery.com">API jQuery</a><br>

<a href="http://docs.jquery.com">Documentation jQuery</a><br>

</div>

<script src="jquery.js"></script>

<script>

$(function() {

var test = $('a').attr('href');

document.write(test);

});

</script>

</body>

</html>

## Modifier les attributs :

## Lire ou ajouter :

* $('#plus').attr('src'); retourne l'attribut src de l'élément d'identifiant plus.
* $('div').attr('class'); retourne l'attribut class du premier <div>.
* $('div').attr('class', 'madiv'); modifie ou crée l'attribut class dans les balises <div> du document et leur affecte la valeur « madiv ».
* $('#illustration').attr('src','monimage.jpg'); modifie ou crée l'attribut src dans la balise d'identifiant illustration et lui affecte la valeur « monimage.jpg ».

## Supprimer :

$(sel).removeAttr('attribut');

On peut aussi accéder aux attributs css :

Getter :

var taille = $('.para').css('font-size');

Setter :

$('.para').css('font-size', '40px');

## Travailler avec l’attribut class :

* addClass() ajoute une classe dans les éléments sélectionnés ;
* removeClass() supprime (si elle existe) une classe des éléments sélectionnés ;
* toggleClass() accomplit deux actions : si la classe spécifiée n'existe pas dans les éléments sélectionnés, elle y est ajoutée. Si elle existe, elle est supprimée.

Attention aux classes qui se chevauchent ! On ne peut pas définir une classe en rouge si elle est déjà écrite en verte :

$('#eric').removeClass('vert').addClass('rouge');

$('#paul').removeClass('vert').addClass('rouge').removeClass('grand').addClass('petit');

Peut aussi être simplifié en :

$('#pierre').removeClass('vert grand').addClass('rouge petit');

## Test de l’existence d’une classe :

L'instruction $('#jean').hasClass('rouge'); renverra la valeur true, car le <span> est de classe rouge.

L'instruction $('#jean').hasClass('petit'); renverra la valeur false, car le <span> n'est pas de classe petit.

On peut donc utiliser des ‘if’ :

if ($('#jean').hasClass('rouge'))

Pour tester l’appartenance à plusieurs classes on utilise : is(). Renvoie true s’il possède bien toutes les classes sélectionnées !

$('#jean').is('.grand.rouge');

## Travailler avec les formulaires :

Vous utiliserez la méthode val() pour tester/modifier la valeur des zones de texte, boutons radio, cases à cocher, listes déroulantes et zones de liste contenues dans un document HTML.

| **Instruction jQuery** | **Effet** |
| --- | --- |
| $('#nom').val() | Lit le nom de l'utilisateur. |
| $('#pass').val() | Lit le mot de passe. |
| $(':radio#H:checked').val() | Lit l'état du bouton radio H. Renvoie true si le bouton est sélectionné, sinon false. |
| $('#fonction').val() | Lit l'élément sélectionné dans la liste déroulante. |
| $('#nom').val('Michel') | Écrit « Michel » dans la zone de texte Nom d'utilisateur. |
| $('#pass').val('abcde') | Écrit « abcde » dans la zone de texte Mot de passe. |
| $(':radio').val(['H']); | Sélectionne le bouton radio H. |
| $('#fonction').val('retraite') | Sélectionne Retraité dans la liste déroulante. |

## Travailler avec des valeurs stockées :

* text() retourne/modifie la valeur textuelle stockée dans l'élément ;
* html() retourne/modifie le code HTML stocké dans l'élément.

$('p').text(function(index,actuel) {

alert('Paragraphe ' + (index+1) + ' : '+actuel);

});

| **Instructions** | **Résultat** |
| --- | --- |
| alert($('p').html()); | Affiche le code HTML du premier élément (voir image suivante). |
| alert($('p:first').html()); | Affiche le code HTML du premier élément. |
| alert($('p:last').html()); | Affiche le code HTML du dernier élément. |
| $('p').html(function(index,actuel) { alert('Paragraphe ' + (index+1) + ' : '+actuel);}); | Affiche individuellement le code HTML de chaque élément. |

Comparé à text(), html() se contente du premier élément et ne les balaie pas tous comme text(). Donc les deux premiers résultats sont identiques.

On peut les utiliser en setter :

$('p:first').html('<a href="http://www.google.com">Moteur de recherche Google</a>');

## Position et taille des éléments :

### Modifier la position :

Pour connaitre et modifier la position d’un élément :

* offset() : position absolue d'un élément dans la page (getter et setter) ;
* position() : position relative d'un élément dans son parent (getter seulement).

Les positions retournées ont deux composantes : abscisse left et ordonnée top, on fera donc :

* offset().left et offset().top pour connaître la position absolue d'un élément.
* position().left et position().top pour connaître la position d'un élément dans son parent.

Prenons le cas de deux div imbriqués, avec un parent et un enfant. On a les classes css suivantes :

#parent {

width: 300px;

height:300px;

position: absolute;

top: 100px;

left: 200px;

background-color: yellow;

}

#enfant {

width: 100px;

height:100px;

position: absolute;

top: 150px;

left: 100px;

background-color: red;

}

Pour #enfant, top et left sont bien en absolu mais relative par rapport au parent !

Pour connaitre les positions on fera donc :

var posparent=$('#parent').offset();

var posenfant=$('#enfant').offset();

$('span').text('Parent : x=' + posparent.left + ', y=' + posparent.top + ' Enfant : x=' + posenfant.left + ', y=' + posenfant.top);

Si on remplace offset() par position () :

Les coordonnées renvoyées sont relatives au parent de chaque balise. Le parent de la balise #parent est le document. Elles ne devraient donc pas changer. Quant au parent de la balise #enfant, il s'agit de la balise #parent. Ses coordonnées seront donc relatives à cette balise.  
Une fois le code modifié et sauvegardé, exécutez le fichier. Vous devriez obtenir l'image suivante.

### Modifier la taille :

var posenfant = $('#enfant').offset();

posenfant.top = 100;

posenfant.left = 100;

$('#enfant').offset(posenfant);

### Connaitre les dimensions :

Plusieurs méthodes jQuery permettent de connaître les dimensions et de redimensionner les éléments de type block :

* width() : largeur de l'élément, de la fenêtre ou du document, sans inclure les marges (padding, border et margin). Cette méthode peut être utilisée comme getter (pour connaître la largeur d'un élément) ou comme setter (pour modifier la largeur d'un élément).
* innerWidth() : largeur de l'élément, en incluant le padding gauche et droit.
* outerWidth() : largeur de l'élément, en incluant le padding gauche et droit et border.
* outerWidth(true) : largeur de l'élément, en incluant padding gauche et droit, border et margingauche et droit.
* height() : hauteur de l'élément, de la fenêtre ou du document, sans inclure les marges (padding, border et margin). Cette méthode peut être utilisée comme getter (pour connaître la hauteur d'un élément) ou comme setter (pour modifier la hauteur d'un élément).
* innerHeight() : hauteur de l'élément, en incluant le padding supérieur et inférieur.
* outerHeight() : hauteur de l'élément, en incluant border et padding supérieur et inférieur.
* outerHeight(true) : hauteur de l'élément, en incluant border, padding supérieur et inférieur et margin supérieur et inférieur.

Pour fixer les dimensions du div, on utilie les setters :

$('div').width('400px');

$('div').height('200px');

### Associer des données aux balises :

Utilisée en setter, la méthode jQuery $.data() permet d'associer une ou plusieurs données textuelles à n'importe quel élément du DOM.

$.data(el, 'nom', nom\_don: don);

… où :

* el est le nom de l'élément concerné, sans apostrophes ;
* nom est le nom (entre apostrophes) de la « variable » dans laquelle sera stockée la donnée ;
* nom\_don est le nom (sans apostrophes) associé à la donnée ;
* don est une donnée quelconque. S'il s'agit d'un nombre, les apostrophes sont inutiles. Par contre, s'il s'agit d'une chaîne, mettez-la entre apostrophes.

Si vous voulez associer plusieurs données à un élément, vous utiliserez une syntaxe légèrement différente :

$.data(el, 'nom', {nom\_don1: don1, nom\_don2: don2, nom\_don3: don3, etc.});

Pour retrouver une donnée :

var uneVariable = $.data(el,'nom').nom\_don;

Pour retirer les données :

var div = $('div')[0];

$.removeData(div, 'mesValeurs');

# Insérer et remplacer des éléments du DOM

## Insérer du contenu

Plusieurs méthodes très pratiques permettent d'insérer du contenu dans ou en dehors de la sélection (entendez par là des éléments retournés par le sélecteur jQuery) :

* append() insère du contenu à la fin de la sélection ;
* prepend() insère du contenu au début de la sélection ;
* before() insère du contenu avant la sélection ;
* after() insère du contenu après la sélection.

## Remplacer du contenu :

$('hr').replaceWith('<br>');

Pour remplacer un titre, on ne remplacerait que <h2>, </h2> va donc disparaitre et il n’y aura aucun rendu.

$('h2').each(function(){

var elemH2 = $(this);

elemH2.replaceWith('<h3>' + elemH2.text() + '</h3>');

});

## Insérer des éléments :

Insérer des éléments dans l’arborescence du DOM :

* eai.appendTo(cible) insère un élément à la fin de la cible ;
* eai.prependTo(cible) insère un élément au début de la cible ;
* eai.insertBefore(cible) insère un élément avant la cible ;
* eai.insertAfter(cible) insère un élément après la cible.

Où eai représente l’élément à insérer.

Ex :

$('<li>Deuxième élément bis</li>').insertAfter($('li:nth-child(2)'));

Ex :

$('<li>Deuxième élément bis</li>').insertBefore($('li:nth-child(2)'));

## Déplacer du contenu :

Pour déplacer un élément existant dans le document, vous utiliserez les méthodes append(), prepend(), before() ou after() :

* $('sel').append(depl);
* $('sel').prepend(depl);
* $('sel').before(depl);
* $('sel').after(depl);

… où sel sélectionne l'élément avant ou après lequel doit se faire le déplacement et depl représente l'élément à déplacer.

## Dupliquer des éléments :

On fait comme pour le déplacement mais on rajoute clone() :

Ex :

$('#deux').clone().insertBefore($('#un'));

Quand on duplique, tous les fils sont dupliqués avec. Ainsi pour faire un sommaire on peut faire :

$('<h1>Sommaire</h1>').insertBefore($('h2:first'));

$('h2').clone().insertAfter($('h1'));

## Entourer des éléments :

La méthode wrap permet d’entourer les éléments.

$('sel').wrap('elwrap');

Autre exemple, si on veut afficher en rouge, gras italique et souligné :

$('li').wrap('<font color="red"><b><i><u></u></i></b></font>');

La méthode wrap() a deux variantes :

* wrapInner(), pour entourer le contenu d'un élément par un autre élément créé à la volée ;
* wrapAll(), pour entourer d'une façon globale les éléments sélectionnés avec un autre élément créé à la volée.

Exemple :

Si on a :

<p>Le texte du paragraphe</p>

Qu’on applique :

$('p').wrapInner('<i></i>');

On aura :

<p><i>Le texte du paragraphe</i></p>

Alors que si on avait juste utilisé wrap normal, on aurait eu :

<i><p>Le texte du paragraphe</p></i>

## Supprimer des éléments :

On utilise la méthode remove().

Ex :

$('h2').remove();

Ou encore : supprimer tous les paragraphes qui contiennent le mot quelconque :

$('p').remove(':contains("quelconque")');