Programmation Web client riche

Année : **2016-2017**

Auteur : **Nassim Hadj-Rabia**

2ème partie

JQUERY

JQuery

- Une bibliothèque javascript open-source et multi-navigateur.
- permet de traverser et de manipuler très facilement l'arbre DOM.
- permet de changer/ajouter une classe CSS, créer des animations, modifier des attributs, etc.
- permet de gérer les événements javascript
- permet de faire des requêtes AJAX simplement

une simple bibliothèque à importer

- bibliothèque en local (http://jquery.com)
  <script type="text/javascript" src="jquery-1.11.3.min.js"></script>

- directement sur Google code

- **la librairie minimale**

  <script type="text/javascript"
  src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js">
  </script>
  ***avantages** : Le script est centralisé et évite de le retélécharger le scrit*
  Bande passante allègée

- **les librairies complètes**

  <script type="text/javascript" src="http://www.google.com/jsapi"></script>

La fonction jQuery()

- jQuery repose sur une seule fonction javascript: **jQuery() ou $().**
- Elle accepte des paramètres et retourne un objet jQuery. Les paramètres peuvent être :

  - une fonction, qui sera exécutée dès que le DOM est disponible.

  - un sélecteur CSS : un objet jQuery correspondant à cet (ces) élément(s) est retourné

  - un sélecteur jQuery spécifique : un objet jQuery est retourné

- **$(document)** représente le document au sens du DOM
- **$(window)** représente la fenêtre au sens du DOM

Sélecteur Basiques

*	**Toutes** les balises
#titre	La balise ayant pour **id** titre
h1	Les **balises** h1
.gras	Les balises qui ont la **classe** *gras*
A, h1,h2	Les balises a, h1 et h2

Exécution du jQuery

- par défaut, exécution du jQuery lors du chargement
- problème : tous les éléments de la page ne sont pas accessibles dans l'arbre DOM (même après fermeture de la balise body)

```
$(function($){
// corps de la fonction
})(jQuery);
```

```
$(function(){
// corps de la fonction
});
```

```
$(document).ready(function(){
// corps de la fonction
});
```

Structure des objets Jquery

Supposons le code suivant :

```
<p> coucou ! </p>
<p> hello ! </p>
<p> hola ! </p>
```

$('p') retourne un objet jQuery qui "contient" 3 éléments **p**
$('p').length donne le nombre d'éléments concernés

Pour accéder aux objets représentant les éléments contenus dans l'objet jQuery

$('p').eq(i), 0 correspond au premier élément

Pour accéder aux noeuds javascript

$('p').get(i)
ou
$('p')[i]

0 correspond au premier élément

Attention, ce ne sont pas des objets jQuery mais des nœuds javascript !

window.alert($('p').get(0).textContent)

Les méthodes

- méthodes de jQuery : des fonctions comme les autres

```
chaine = $.trim(chaine);
```

- méthodes des objets jQuery

```
$("sélecteur").methode(paramètres);
```

- Utilisation de fonction anonyme

```
$(document).ready(function(){
var text = $.trim(" mon texte  ");
// fonction alert() native de javascript
window.alert(text);
});
```

Différence text() et html()

- **html(chaine)** : **remplacement** du contenu d'un élément (les **balises** sont considérées comme des **balises et non ue chaine de caractères**)

  $("p").html("Hello world!");

  => *remplace le contenu de l'élément **p** par l'élément Hello* **world !**

- **html()** : récupération du contenu d'un élément

  $('p').html() => *"Hello world!"*

- **text(chaine)** : remplacement du contenu d'un élément en considérant le tout comme du texte (les caractères < et > des balises sont remplacés par les entités > et <)

  $('span').text(" il est < 5") => *remplace le contenu de l'élément span par "il est &lt ; 5"*

  http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_text_set

- **text()** : récupération du contenu d'un élément sous forme de texte

  $('span').text() => il est < 5

Itérations

each() est une méthode qui permet d'itérer sur les éléments contenus dans un objet *jQuery*

la fonction anonyme passée en argument est appelée autant de fois qu'il y a d'éléments dans l'objet *jQuery*. Si cette fonction retourne false, **each()** arrête d'itérer.

this dans la fonction représentera l'objet jQuery concerné par l'itération. La variable **i** représente la variable qui s'incrémente à chaque itération.

```
$('a').each(function(i) {
$(this).append('<span>Lien n°'+(i+1)+'</span>')
});
```

Remarque : peut aussi s'écrire

```
$('a').append(function(i){
    return '<span>Lien n°'+(i+1)+'</span> ';
  })
```

Sélecteurs CSS

- **similaires aux éléments d'une feuille de style**

  ex : #menuid, h2, .onglet, *, ...

  Référence:
  http://docs.jquery.com/DOM/Traversing/Selectors#CSS_Selectors

- **Exemple (ces fonctions retournent un objet JQuery)**

  $('body'), correspond à l'élément **body**

  $('a[href]'), correspond aux éléments **a** ayant un attribut **href**

  $('a[href^=http://'), correspond aux éléments **a** ayant un attribut **href** dont la valeur commence par *http://*

  $('img[src$=.png'], correspond aux éléments **a** ayant un attribut **src** dont la valeur finit par *.png*

  $('div,p')**, correspond aux éléments **div** et **p

Quelques sélecteurs CSS

sélecteur CSS	retour fonction $ (objet jQuery contenant)
elem	balises elem
#ident	balise ayant l'id "ident"
.classe	balises ayant la classe "classe"
elem[attr]	balises elem dont l'attribut "attr" est spécifié
elem[attr="val"]	balises elem dont l'attribut "attr" est à la valeur val.
elem bal	balises bal contenues dans une balise elem
elem > bal	balises bal directement descendantes de balises elem
elem + bal	balises bal qui **vient après** chaque balise elem
elem ~ bal	balises bal précédées d'une balise elem (au sens frère)

Exemple (1)

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

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_sel_previous_next

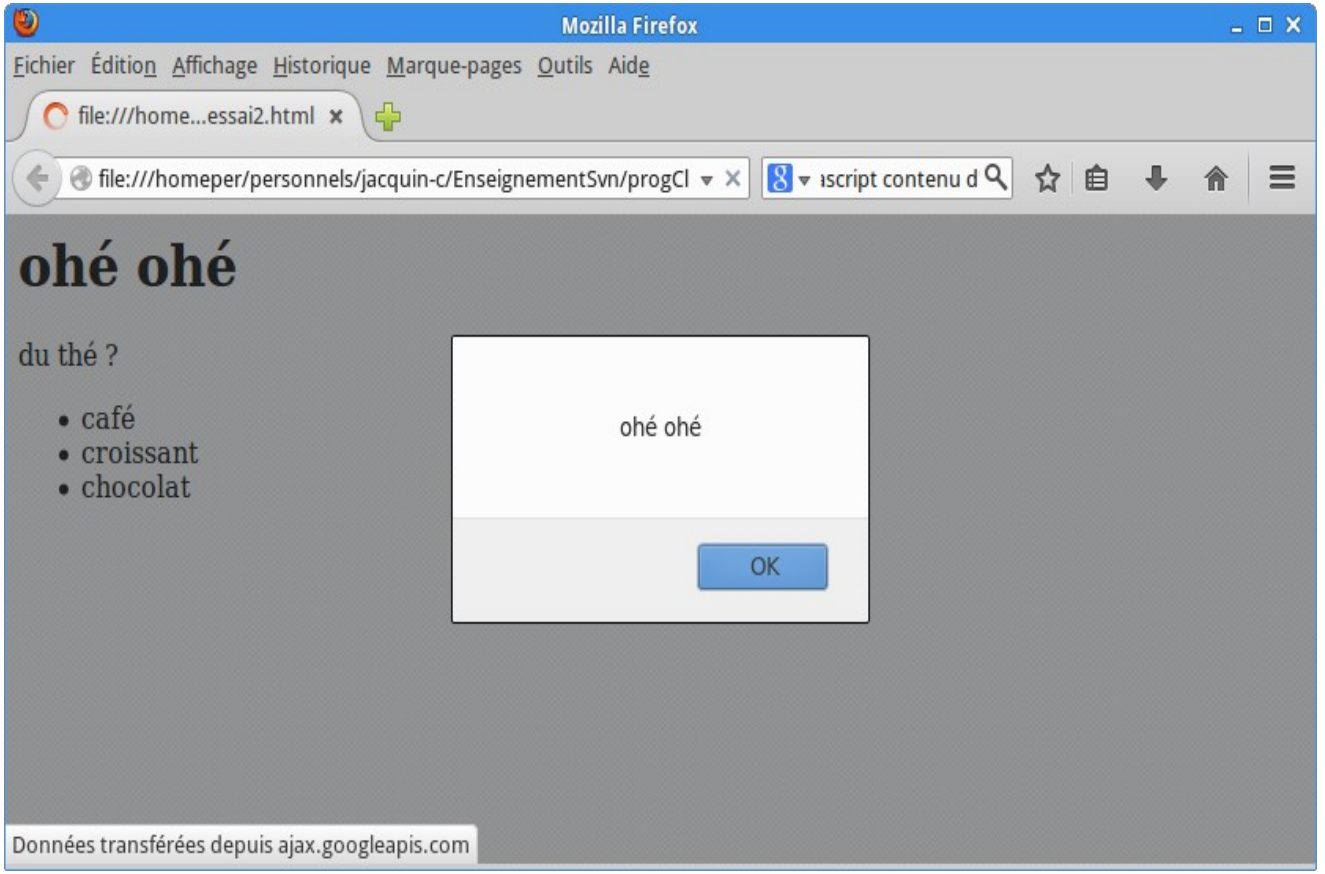
- $("div ~ p").css("background-color", "yellow");

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_sel_previous_siblings

Exemple (2)

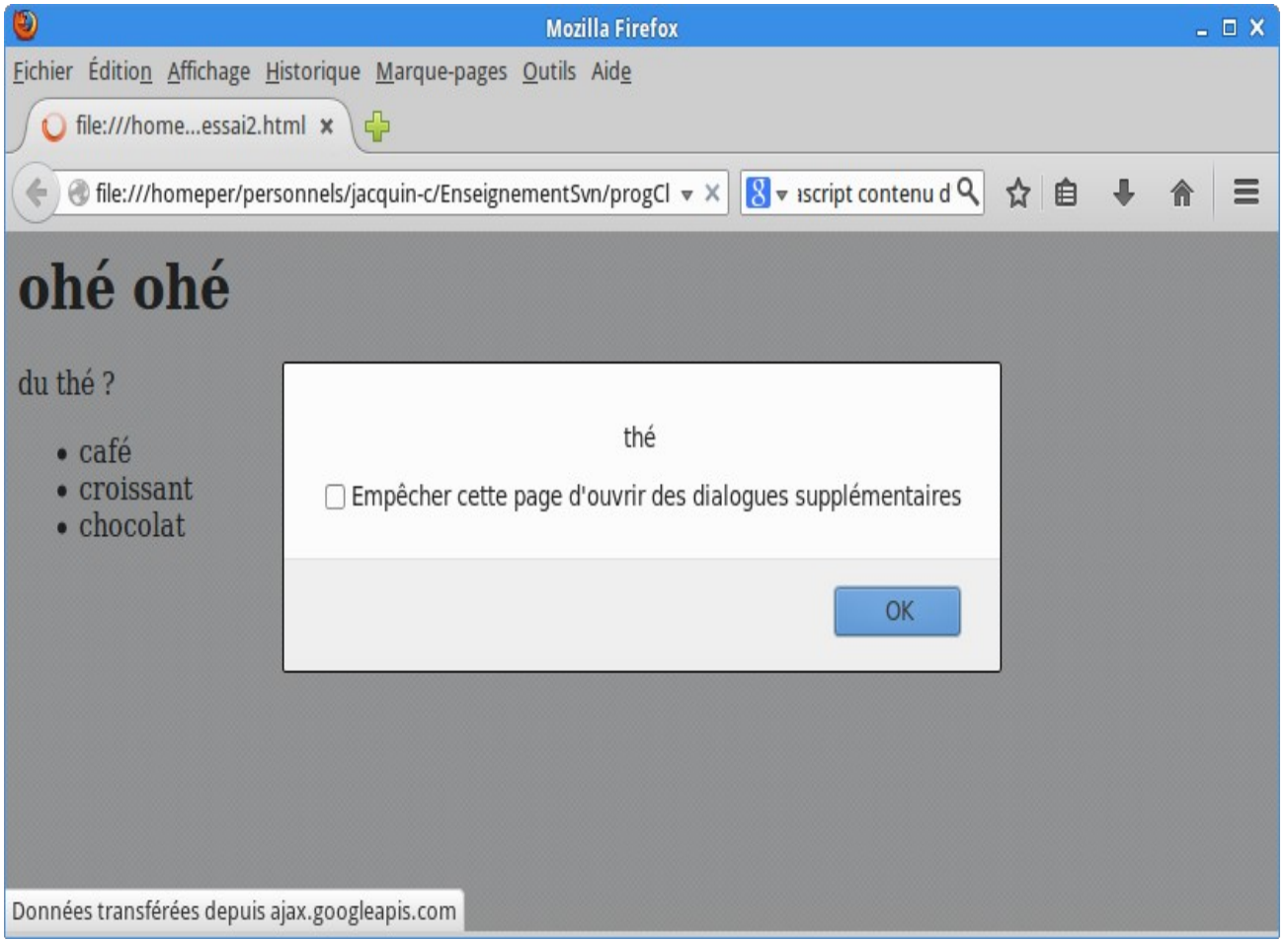
```
<body>

salut tout le monde !

<h1> ohé ohé </h1>

<span>du <span> thé </span> ? </span>

<ul><li id ="li1"> café </li><li> croissant </li><li> chocolat</li>

</ul>

<script type="text/javascript">

    /* la fonction html() appliquable à des objets jQuery permet de récupérer le
    contenu d'une balise la fonction html(chaine) permet d'écrire chaine dans la
    balise sélectionnée. */

    window.alert($('h1').html());

    window.alert($('span').eq(1).html());

    $("#li1").html($('span').eq(1).html());

    $("span").html("");

    </script>

</body>
```

Exemple (2)



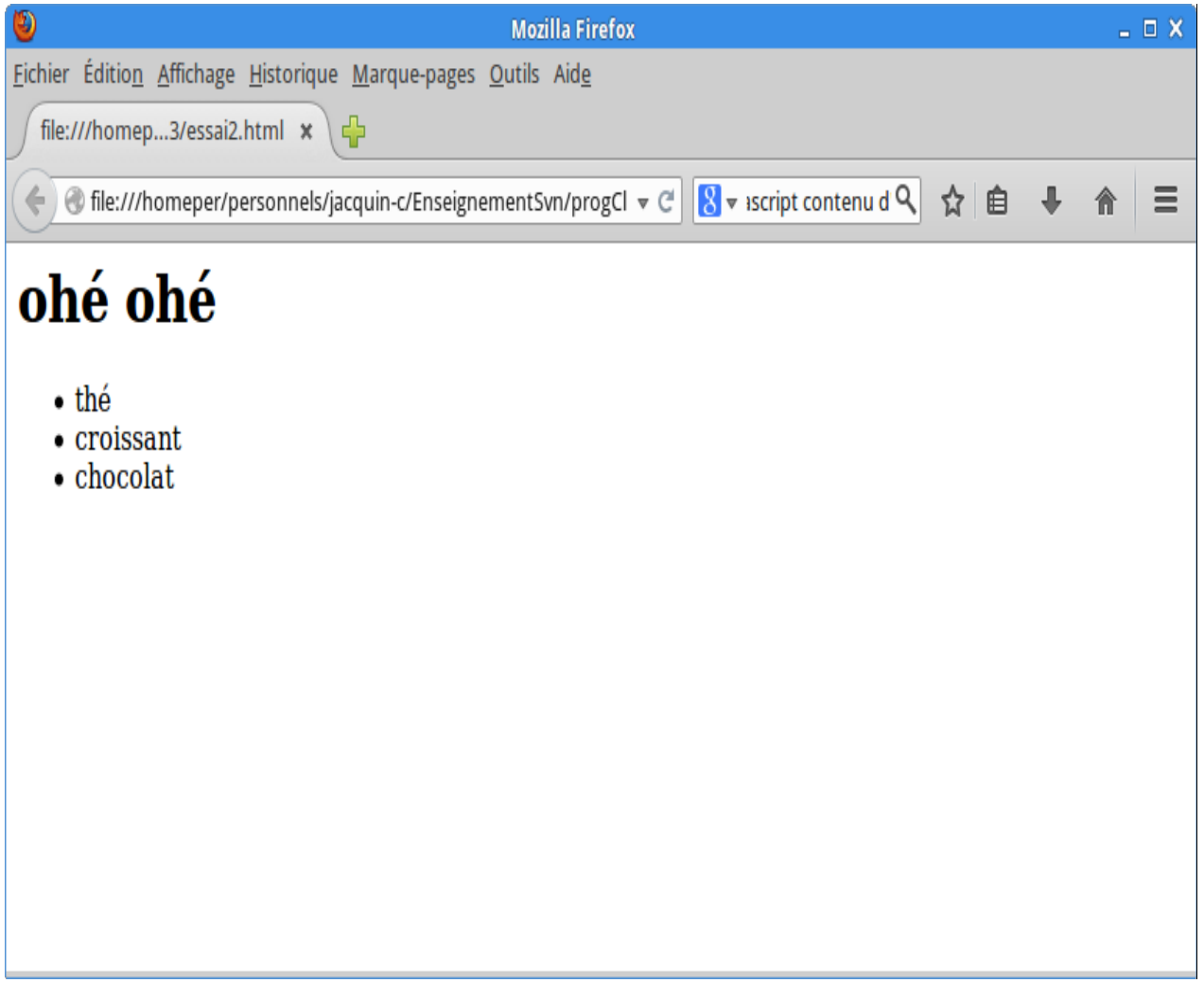

```
<body>
salut tout le monde !

<h1> ohé ohé </h1>
<span>du <span> thé </span> ?
</span>

<ul>
<li id ="li1"> café </li>
<li> croissant </li>
<li> chocolat</li>
</ul>

<script type="text/javascript">
window.alert($('h1').html());
window.alert($('span').eq(1).html());
$("#li1").html($('span').eq(1).html());
$("span").html("");
</script>
</body>
```

Exemple (3)




```
<body>
salut tout le monde !

<h1> ohé ohé </h1>
<span>du <span> thé </span> ?
</span>

<ul>
<li id ="li1"> café </li>
<li> croissant </li>
<li> chocolat</li>
</ul>

<script type="text/javascript">
window.alert($('h1').html());
window.alert($('span').eq(1).html());
$("#li1").html($('span').eq(1).html());
$("span").html("");
</script>
</body>
```

Exemple (4)




```
<body>
salut tout le monde !

<h1> ohé ohé </h1>
<span>du <span> thé </span> ? </span>

<ul>
<li id ="li1"> café </li>
<li> croissant </li>
<li> chocolat</li>
</ul>

<script type="text/javascript">
window.alert($('h1').html());
window.alert($('span').eq(1).html());
$("#li1").html($('span').eq(1).html());
$("span").html("");
</script>
</body>
```

Sélecteurs spécifiques JQuery

- **des sélecteurs en forme de filtres:**

  :checked, :odd ,:visible ,:contains(du texte),:button, ...

exemples: $('img:visible') => tous les éléments **img** visibles

$(:button) => tous les éléments de type **button**

Référence:
http://docs.jquery.com/DOM/Traversing/Selectors#Custom_Selectors

Quelques sélecteurs jQuery

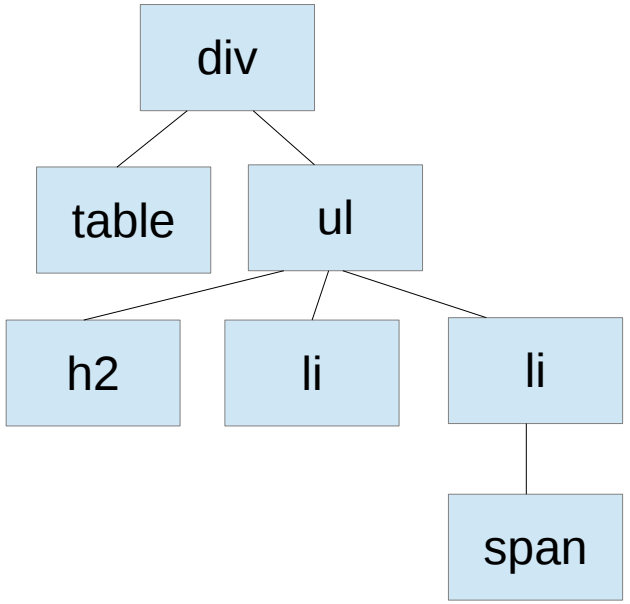
Sélecteurs jQuery	Retour de la fonction $ (un objet jQuery contenant)
:hidden	éléments invisibles
:visible	éléments visibles
:parent	éléments qui ont des éléments enfants
:header	balises de titres : h1, h2, h3, h4, h5 et h6
:not(s)	éléments non sélectionnés par le sélecteur s
:has(s)	éléments contenant des éléments sélectionnés par le sélecteur s
:contains(t)	éléments qui contiennent le texte t
:empty	éléments dont le contenu est vide
:eq(n)	le n-ième élément, en partant de zéro
:gt(n)	éléments dont l'index est plus grand que n
:lt(n)	éléments dont l'index est plus petit que n
:first	premier élément (équivaut à :eq(0)).
:last	le dernier élément
:even	éléments dont l'index est pair
:odd	éléments dont l'index est impair

Les méthodes associées aux objets jQuery

- les méthodes s'appliquent à tous les éléments sélectionnés
  - $('.classe').hide();
  - $('.classe').show();
- de nombreuses méthodes utilitaires
  - parcourir le DOM: parent(), next(), children(), parents(), siblings(), find()
  - ajouter ou retirer des classes CSS: addClass, removeClass
  - manipuler: append, wrap, prepend
- Intérêt fondamental: la plupart des méthodes de l'objet retournent l'objet lui-même
  - on peut ainsi chaîner les appels

    $('anything').parent().find('still anything').show();

Naviguer dans le DOM



$('ul').parent() => l'élément *div*

$('span').parents() => l'élément *li* (de droite), *ul* et *div*

$('ul').children() => les 2 éléments *li et h2*

$('h2').next() => l'élément *li (de gauche)*

$('h2').siblings() => les 2 éléments *li*

$('div').find('span') => l'élément *span*

Autre exemple:

$("div").children("ul.maliste") => les fils directs de div qui sont des éléments de type *ul* et de classe *maliste*

http://www.w3schools.com/jquery/trysel.asp

Remplacement d'éléments

- replaceWith(): remplacement de l'élément **courant** (balise et contenu) par l'élément en paramètre

```
$("p:first").replaceWith("<h2>Hello world!</h2>");
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_replacewith

- replaceAll() : remplacement de l'élément en **paramètre** (balise et contenu) par l'élément sur laquelle est appliquée la méthode

```
$("<h2>Hello world!</h2>").replaceAll("p");
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_replaceall

Déplacement d'éléments

A l'intérieur :

- **prepend(param)** (ajout élément représenté par ***param*** comme 1er enfant)
- **append(param)** (ajout élément représenté par ***param*** comme dernier enfant)

A l'extérieur :

- **before(param)** (ajout élément représenté par ***param*** avant l'élément ciblé)
- **after(param)** (ajout élément représenté par ***param*** après l'élément ciblé)

- param peut être soit une chaîne de caractère ou un objet jQuery

- si **param** est un objet jQuery, il n'est pas cloné mais déplacé

Exemple de déplacement

Prepend :

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_prepend

Append :

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_append

Before et After :

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_after

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_after2

Ajout et suppression d'éléments

- ajout de balises

  à l'extérieur : wrap(), wrapAll()

  ```
  $("p").wrap("<div></div>");
  ```

  http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_wrap

  à l'intérieur : wrapInner()

  ```
  $("p").wrapInner("<b></b>");
  ```

- suppression de la balise parent : unwrap()

  ```
  $('#id4').unwrap();
  ```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_unwrap

Manipulation d'éléments

- copie d'élément : clone()

```
$('p').clone().appendTo("body") ;
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_clone

- suppression d'éléments: remove()

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

- suppression de tous les sous-éléments: empty()

```
$('#id3').empty();
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_dom_remove

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_dom_empty

Quelques méthodes et attributs utiles

- **variable tagName** : nom de l'élément DOM en majuscule
- **méthodes permettant de manipuler les attributs des éléments**

  - *attr(attribut)* : récupération d'un attribut de l'objet DOM courant

  - *attr(attribut,valeur)* : association d'une valeur à un attribut de l'objet DOM courant

  - *removeAttr(attribut)* : suppression de l'attribut de l'objet DOM courant

- **exemple: déterminer si une checkbox est cochée**

```
if ($('#maBoite').attr('checked')) {
        //traitement si cochée }
        else {
        //traitement si non cochée}
```

Exemple du tri d'un tableau suivant les éléments de sa première colonne

```
$tBody = $("table[id=1] tbody") ;

/* Capture des lignes du tableau
et on passe en paramètre à sort la fonction de
comparaison entre 2 éléments */

$tBody.children("tr").sort(function(ligneA,ligneB) {
// tri suivant l'élément de la 1ère colonne du tableau

var valeurA = $(ligneA).children('td:first').text();

var valeurB = $(ligneB).children('td:first').text();

return valeurA > valeurB ? 1 : -1 ;

})

// Les lignes sont ajoutées à $tBody dans l'ordre de tri

.appendTo($tBody) ;
```

```
<table id= "1">
...
<tbody>
<tr><td>tisane</td><td> 12 </td></tr>
<tr><td>café</td><td>3</td></tr>
<tr><td>thé </td><td>45</td></tr>
<tr><td>chocolat</td><td>6</td></tr>
</tbody>
</table
```

La gestion des événements en jQuery (depuis version 1.7)

association d'un(de) gestionnaire(s) d'événement à un objet jQuery par la méthode on()

- $('p').on('**mouseover**', function(){
  $(this).css('background-color','#FFF') ;
  })

association de plusieurs gestionnaires d'événement avec un comportement différent

$('p').on(
mouseover: function(){
$(this).css('background-color','#FFF') ;} ,
click: function(){
$(this).css('background-color','#AAA') ;

});

- suppression du gestionnaire d'événement associé à un objet jQuery

$('a').off('click') ;

La délégation

- **délégation d’événements**

  définir un gestionnaire d'événement pour tous les éléments d'un certain type (qui peuvent être créés dynamiquement)

```
$('div').on('click','p', function()
{$(this).css('background-color','#FFF') ;
}) ;
```

Ici tous les éléments **p**, descendant de **div** auront ce gestionnaire d'événement qui leur sera attaché.

La délégation

- **délégation d’événements**

  définir un gestionnaire d'événement pour tous les éléments d'un certain type (qui peuvent être créés dynamiquement)

```
$('div').on('click','p', function()
{$(this).css('background-color','#FFF') ;
}) ;
```

Ici tous les éléments **p**, descendant de **div** auront ce gestionnaire d'événement qui leur sera attaché.

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_event_delegate

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_hide

Avantage de la délégation (1)

```
<div id="div1">
<ul>
<li>
<img src="http://www.univ-nantes.fr/iut.png" alt=""/>
<a href="http://www.google.com">More info</a>
<button>appuyer</button>
</li>
...
</ul>
</div>
```

- les gestionnaires d'événement associés

```
$("#div1").on('click', function(){ ….}) ;
$("img").on('click', function(){ ….}) ;
$("a").on('click', function(){ ….}) ;
$("button").on('click', function(){ ….}) ;
```

- **Problème** : le temps de chargement de tous les gestionnaires peut être long !!!!

Avantage de la délégation (2)

```
var target=event.target ;
$("#div1").on('click', 'img, a, button', function(event){
switch(target.tagName.toLowerCase()) {
    case 'img':
      ...
      break;

    case 'a':
      …
         break ;
        case 'button':
        …

        break ;
}}) ;
```

```
<div id="div1">
<ul>
<li>
<img src="http://www.univ-nantes.fr/iut.png" alt=""/>
<a href="http://www.google.com">More info</a>
<button>appuyer</button>
</li>
...
</ul>
</div>
```

résolution du problème de temps de chargement (1 seul gestionnaire d'événement chargé) !

Les formulaires

les sélecteurs	désignation
:checked	les checkbox cochées ou boutons radio sélectionnés
:selected	les options (provenant de select) sélectionnés.
:disabled	les éléments de formulaires désactivés.
:enabled	les éléments de formulaires actifs.
:input	tous les éléments d'un formulaire
:button	tous les boutons
:reset	tous les boutons reset
:submit	tous les boutons submit
:checkbox	toutes les cases à cocher
:radio	tous les boutons radio
:text	tous les champs de texte
:password	tous les champs de mots de passe
:hidden	tous les champs cachés

Les formulaires

- val() permet de renvoyer différents types d'information concernant les formulaires.
- val(chaine) permet de valuer certaines informations dans les champs de formulaire.

```
$('#ok').on('submit',function() {
if ($('#login').val() =="") {
event.preventDefault();
alert ('Entrer un login svp !!!')
}
})
```

Formulaire (exemple 1)

- **effacer le contenu d’un champs de texte lorsqu’il a le focus**

```
<input name=“nom” type=“text” id=“nom” value=“Entrez votre nom”>

$('#nom').on(focus, function() {
var champs = $(this);
champs.val('');
});
```

Formulaire (exemple 2)

- **gérer le clic sur n’importe quel bouton radio**

```
$(':radio').on(click,function() {
//...
});
```

- **donner le focus à un élément d’un formulaire**

```
$('nom').focus();
```

jQuery et CSS

Méthode => **css()**

- **récupérer la valeur d'une propriété**

```
$('p').css('color');
```

- **définir des propriétés css**

```
$('div').css('color','red');
```

```
$('p').css({
color : 'red',
borderColor : 'blue',
paddingRight : '30px',
marginLeft' : '10px'
});
```

- **d'autres méthodes qui permettent de modifier la position des éléments, de gérer leur dimension.**

Bibliographie

- Cours de première année de DUT informatique de l'IUT de Nantes, 2015-2016, Christine Jacquin

- http://www.gchagnon.fr/cours/dhtml/

- Alsacreation, http://www.alsacreations.com/