Framework JavaScript jQuery

api.jquery.com : la référence ! Plein d'exemples

jquery.developpeur-web2.com

visualjquery.com : Excellent

Bibliothèques JavaScript

- Limites de JavaScript
  - Seulement quelques fonctions pour accéder au DOM
  - Animations difficiles à programmer
- D'où, le développement de frameworks pour gagner du temps dans le développement des applications
- Framework → Bibliothèque = ensemble de fonctions
  - simplifiées par rapport à JavaScript seul
  - syntaxe plus simple, plus concise
  - traitement uniforme sur les navigateurs les plus courants

Il faut absolument connaître JavaScript et css afin de maîtriser jQuery

Bibliothèques javaScript

- Prototype, Script.aculous, jQuery
  - + d'autres bibliothèques gratuites qui permettent des manipulations par drag and drop...

- Bibliothèques javaScript offrent des facilités pour
  - Manipuler le DOM
  - Manipuler les CSS
  - Gérer les événements
  - Créer des effets graphiques
  - Faire des requêtes Ajax

Bibliothèque jQuery

http://docs.jquery.com/

http://jquery.developpeur-web2.com/

- Sa devise : « Write less do more »
- jQuery : version 1.10.2
  - J : pour javaScript
  - Query : pour sa facilité à chercher des noeuds dans l'arbre du DOM
- Bibliothèque la plus utilisée : ~45% des sites
  - sur des sites comme Dell, Google Code, Amazon, Mozilla, Wordpress…
- Bibliothèque compatible (n'entre pas en conflit) avec d'autres bibliothèques
- Logiciel libre (licence GPL et MIT)
- Légère : 100Ko
- Communauté très active qui offre de nombreux plugins

Utiliser jQuery

- Pour se servir de la librairie
  - insérer une balise indiquant où trouver la librairie

```
<script type="text/javascript" src="jquery-1.10.2.js"></script>
<script type="text/javascript" src="monFichier.js"></script>
```

  - Tout se fait par l'appel à un objet jQuery
  - Le symbole $(...) est un raccourci pour jQuery
  - Tout script jQuery commence par :

MonFichier.js

```
$(document).ready ( function() {
        /* code jQuery correspondant au traitement à effectuer
           lorsque le DOM est complètement chargé */
        });
```

Base de jQuery : accès aux éléments du DOM

- Une fois l'arbre du document chargé, on est dans la fonction associée à document.ready, on y définit ce qu'on veut faire
  - Sélectionner les éléments du DOM qui nous intéressent **grâce à des sélecteurs**
  - Les manipuler **grâce à des méthodes**

Panorama des simplifications que permet jQuery

Simplification des sélections

```
<div>
    <h2>Je suis un titre, j'ai de l'importance.</h2>
    <p class="rouge">Je suis un paragraphe de classe css rouge</p>
    <p class='onglet'>Ceci est un paragraphe.</p>
    <div class="rouge">Je suis une division de classe css rouge</p>
</div>
<ul id='liste'>
    <li class='onglet'>bonjour</li>
    <li class='onglet'>au revoir</li>
</ul>
```

Sélectionner tous les éléments p ayant pour classe « rouge »

En javaScript : document.getElementsByTagName("p")

puis boucle de recherche de ceux ayant pour classe « rouge »

Avec jQuery, une ligne suffit : `$("p.rouge");` → Notation à la css

Méthodes adressées aux nœuds sélectionnés boucle implicite si sélection multiple

```
$("#liste").addClass("rouge") ;

$('p').addClass("vert") ;

$('p.rouge').removeClass('rouge');

$("#liste .onglet").hide() ;

$("#liste .onglet:first").show();

console.info($('.rouge').length) ;
```

Evénementiel plus simple

```
function affiche(){....}

$("#monForm").submit(affiche);

$("div").click(function(){
                console.info('vous venez de cliquer sur la div !!!');
            }) ;
```

Fonction nommée

Fonction anonyme

Enchainement de méthodes boucle et utilisation de $('this')

```
$('#liste').append('<p> un texte</p>').css("border","1px solid red")

$("img").each(function(){
                console.log($(this).attr("src"));
                }) ;

$("li").hover(
    function () { $(this).css("background-color", "green");
                  },
    function () {$(this).css("background-color", "transparent");
                  }
  );
```

Simplification des formulaires

$(':checkbox')

- Sélectionne toutes les checkbox

$(':checkbox:checked')

- sélectionne toutes les checkbox cochées

$(' :checkbox[name="situation[]"]:checked ') :

sélectionne tous les checkbox cochées du groupe situation

Détail des possibilités de sélections de nœuds

Sélection des noeuds du DOM beaucoup plus facile qu'en javaScript

$(...) retourne un objet jQuery contenant les éléments, répondant à la sélection de nœuds indiquée dans les parenthèses

Sélectionner des éléments du DOM notation proche de css

- $("#menu a")
- $("#menu *")
  - les descendants = enfants, petits enfants, petits-petits enfants...
- $(" li > a") enfants « stricts »
- $('div.rouge') : sélectionne les éléments div ayant la classe "rouge"
- $('div .rouge') : sélectionne les éléments ayant la classe "rouge" descendants (au sens CSS) d'éléments

- On peut utiliser plusieurs sélecteurs à la fois comme en css : **notation ,**
  - $('ul, ol, dl') // Sélectionne les éléments <ul>, <ol> et <dl>
  - $('#menu a, div .rouge')

Appliquer des filtres aux noeuds sélectionnés (1)

notation :

- jQuery peut sélectionner des éléments en appliquant des filtres sur les éléments sélectionnés
  - $('div:first') Sélectionne le premier élément
  - $('div:last') Sélectionne le dernier élément
  - $('div:even') Sélectionne les div paires
  - $('div:odd') Sélectionne les div impaires
  - $("p:eq(4)"); Sélectionne le quatrième paragraphe
  - $("p:lt(8)"); (lt : less than, gt) Sélectionne les huit premiers paragraphes
- On peut **nier** un filtre avec **not**
  - $('div:not(.rouge)') Sélectionne les n'ayant pas la classe "rouge"
- La notation `[a | b]` signifie un 'ou' entre les filtres a, b
  - $('p:[first | last]') Sélectionne le premier paragraphe et le dernier

Appliquer des filtres aux éléments sélectionnés (2)

- $('div p:first-child') // Les paragraphes qui sont les premiers enfants des éléments
- $('div p:last-child')
- $('div p:nth-child(2)') // Sélectionne les p qui sont les 2ème enfants des div
  - Commence à 1
- $('div:empty') //Sélectionne les éléments vides
- $("p:contains('toto')"); // Sélectionne les paragraphes qui contiennent toto

À éviter car recherche de chaînes de caractères

$('div:has(p)') // Sélectionne les div qui contiennent des p

Appliquer des filtres sur les attribut (3)
notation crochets [nomAttribut]

- Récupérer tous les éléments ayant un attribut donné
  - $('p[title]') //Sélectionne les éléments <p> ayant un attribut "title".
- Récupérer tous les éléments ayant un attribut de valeur spécifique
  - $('p[title="Bonjour"]') // Sélectionne les éléments <p> dont l'attribut title est "Bonjour".
  - $('p[title!="Bonjour"]') // Sélectionne les éléments <p> dont l'attribut title n'est pas "Bonjour".
  - $('p[title^="H"]') // Sélectionne les éléments dont l'attribut title commence par "H".
  - $('p[title$="H"]') // Sélectionne les éléments dont l'attribut title finit par "H".
  - $('p[title*="H"]') // Sélectionne les éléments dont l'attribut title contient "H".

Filtrage des éléments de formulaire (4)

- filtres spécifiques à la sélection d'éléments de formulaires en fonction **de leur nature**
  - $(':input') // Tous les éléments <input>, <textarea>, <select> et <button>
  - $(':[text|password|radio|checkbox|button]') // Les <input /> du type choisi.
- filtres spécifiques à la sélection d'éléments de formulaires en fonction **de leur état**
  - $(':checked') // tous les <input> dont l'attribut indiqué est à true
  - À la place de checked, on peut mettre aussi

  (‘[enabled|disabled|selected]')
- $(':checkbox:checked) : toutes les checkbox cochées

En jQuery, plus de propriétés des nœuds mais uniquement des méthodes à adresser aux nœuds

JavaScript	JQuery
nd.textContent = 'toto' ;	$(nd).text('toto') ;
ndBut.disabled = 'disabled' ;	$(ndBut).prop('disabled','disabled) ;
nd.onclick = function(){...} ;	$(nd).click(function(){...}) ;
var s = ndInput.value ;	var s = $(ndInput).val() ;
var s = nd.innerHTML ;	var s = $(nd).html() ;
nd.id='div2' ; nd.title='impor' ;	$(nd).prop({id:'div2', title: 'impor''});

Méthodes pour :

1. Accéder à la structure du document
2. Agir sur le style css
3. Définir des effets visuels
4. Modifier la structure du document
5. Définir de l'événementiel

1. accéder à l'arbre

- Accès au père (unique) car arbre DOM : méthode parent()
- Accès à tous les ancêtres : méthode parents()
- Accès à tous les enfants : méthode children()
- Accès à tous les frères : méthode siblings()
- Accès au frère précédent : méthode prev()
- Accès au frère suivant: méthode next()

ajouter un filtrage à ces méthodes d'accès

`$("#toto").siblings(".rouge");` frères du nœud d'identifiant toto qui sont de classe rouges

`$("p").parents("div") ;` les ancêtres des p qui sont des div

`$('#toto').children('p') :` les enfants du nœud d'identifiant toto qui sont des p

2. Agir sur le style CSS

- Ajouter/enlever une classe css aux éléments sélectionnés

```
$("div").addClass("rouge");
$("div").addToggleClass("rouge") ; //ajoute la classe ou la supprime si elle
                                                    est déjà présente
```

- Ajouter/accéder une propriété css

Format json

```
alert ( $('p').css('background-color') ) ;
$('ul li').css('color', 'red') ;
$('ul li:first').css( { color: "green",  textAlign: "center",  cursor: "pointer" } ) ;
```

Les méthodes en jQuery peuvent « cacher » une boucle implicite

- Une méthode peut s'appliquer soit :
  - à l'élément sélectionné s'il est unique

```
var texte = $("#bonj").html();
```

  - à chaque élément du groupe sélectionné et **cache alors une boucle implicite**

```
$('div').addClass("rouge") ;
```

Quelquefois, on a besoin de créer ses propres boucles : on utilise $(this) pour l'élément courant

- html

```
<div>Bonjour à tous ! </div>
<div>au revoir à tous ! </div>
```

- code jQuery

```
$("div").each(function(i){

        var tex = $(this).html();

        $(this).html( " div : " + (i+1)+ tex );
});
```

each(uneFonction) exécute uneFonction sur chaque élément sélectionné i : $(this) est la div courante

Chaînage des méthodes

- La plupart des méthodes de jQuery ont une particularité : elles renvoient la sélection jQuery elle-même. Cela permet de chaîner les méthodes
- au lieu d'écrire

```
$(".toto").addClass("rouge");
$(".toto").removeClass("vert");
```

- on peut simplifier

```
$(".toto").addClass("rouge").removeClass("vert");
```

3. Définir des effets visuels : démo

http://jquery.developpeur-web2.com/demonstration.php

- Montrer/masquer : méthodes show/hide

  $('#effet).hide() ;

- Faire un « fondu » ~ faire disparaître/apparaître un élément avec une vitesse voulue, avec une opacité voulue

  méthodes fadeOut / fadeIn/ fadeTo (joue sur l'opacité entre 0 et 1)

  ```
  $('#effet).fadeOut("slow");
  $('#effet).fadeIn("fast");
  ```

- Jouer sur l'opacité d'un nœud fadeTo (joue sur l'opacité entre 0 et 1)

  $('#effet').fadeTo("slow", 0 .5) ;

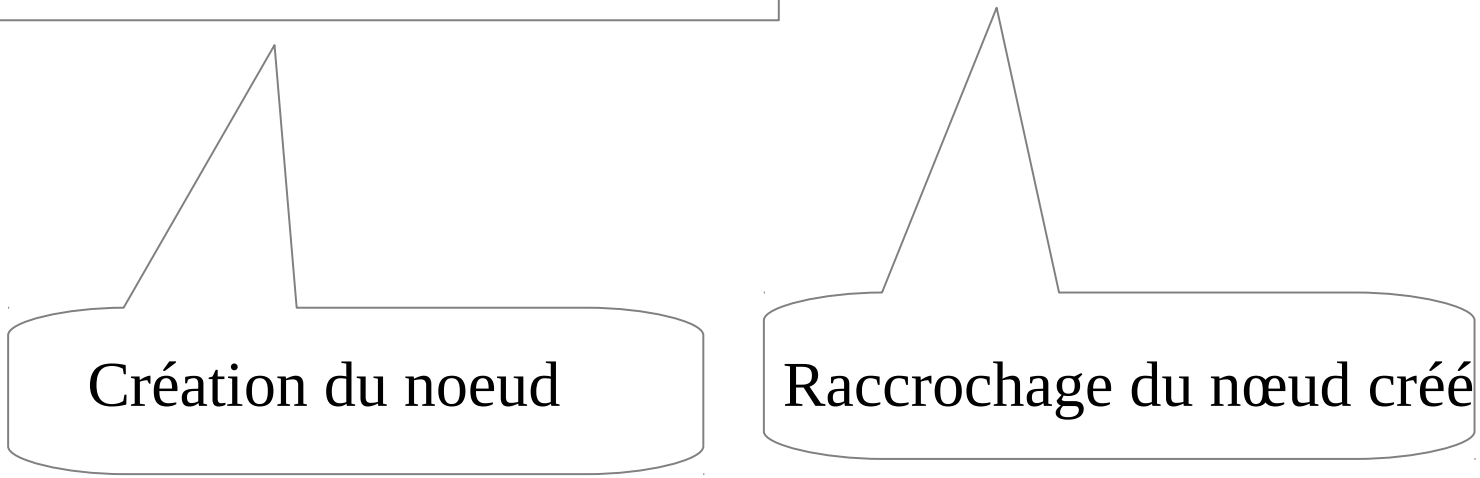
- Faire glisser un élément vers le haut/bas

  méthodes slideUp/slideDown

4. Modifier la structure de l'arbre

4.1 Créer des noeuds

```
$('<p>Un nouveau paragraphe </p>').appendTo($('#unediv')) ;
```



4.2 « Raccrocher » des nœuds à l'arbre existant

- Toute méthode à sa méthode symétrique
- En fils de :
  - Dernier fils : Méthode append/appendTo
  - Premier fils : Méthode prepend/prependTo
- En frère de
  - Méthode before/insertBefore
  - Méthode after/insertAfter
- Ces méthodes s'appliquent à des nœuds existant déjà dans l'arbre ou pas
  - Attention : Si le nœud existe déjà, il est **déplacé** !!!
  - Sinon, il est placé à l'endroit indiqué

4.3 Remplacer/enlever/ « vider » un nœud

```
<div id='unediv'>
        <p>Un nouveau paragraphe</p>
</div>
```

- Déplacer un noeud, remplacer un élément **: méthode replaceWith(...)**

```
$('#unediv p').replaceWith('<h2>New heading</h2>')
```

- Enlever un noeud et tout son sous-arbre : **méthode remove**

```
// enlever tous les noeuds de type p et leurs sous-arbres
$("p").remove() ;          //restera <div id='unediv'></div>
```

- « Vider » un noeud en le laissant : **méthode empty()**

```
$("p").empty() ;
    //restera <div id='unediv'><p></p></div>
```

5. Méthodes de jQuery pour faire de l'événementiel

Associer des événements à des éléments du DOM

- Aux gestionnaires d'événements de javaScript, correspondent des fonctions jQuery
- On enlève le 'on' et c'est une fonction qui s'applique aux éléments sélectionnés
  - onclick -> click()
  - onselect → select()
  - dblclick(), blur(), focus(), select(), keydown(), keyup(), keypress(), mouseover(), mousedown...
    - Ces fonctions prennent en paramètre le nom du callback ou une fonction anonyme

Exemple : répondre au clic

- Html

```
<p id="lien">Dis bonjour !</p>
```

- Associer à ce noeud une réaction au clic
  - Avec une fonction **nommée**

```
$("#lien").click( bonjour) ;
```

Nom d'une fonction

  - **Avec une fonction anonyme**

```
$("#lien").click( function(){
                        alert("bonjour");
                    }
                );
```

Utiliser l'événement

- L'événement est un objet javaScript passé en paramètre de la fonction de callback
- On peut l'utiliser pour accéder à ses propriétés, pour lui adresser des méthodes

```
<a id='lab'   href='http ;//lipn.fr'>Labo </a>

$('#lab').click(function(event){
                    ....
        event.preventDefault() ;
}) ;
```

annule le traitement par défaut lors du clic sur un lien afin que le navigateur ne suive pas la cible du lien

Supplément

Réduire la taille du code et obtenir de meilleures performances : utilisation des méthodes *find* et *end*

- Les sélections nécessitent des parcours d'arbres coûteux en temps : **ne jamais faire plusieurs fois la même sélection**

```
$('div.rouge li').addClass('rouge') ;
$('div.rouge').addClass('italique') ;
```

- A remplacer par :

```
$('div.rouge').addClass(italique').find('li').addClass('rouge');
```

ou

```
$('div.rouge').find('li').addClass('rouge').end().addClass(italique') ;
```

- La méthode *end* ....
- La méthode *find* ...

Du supplément pratique

- On peut faire des boucles en jQuery

```
var t=[2,7,8];
$.each( t, function()   {console.info($(this)); }   );
```

Tableau ou objet
sur quoi porte la boucle

Fonction à appliquer
à chaque élément

Utiliser des plugin : une autre force de jQuery

- Communauté importante qui réalise des plugins
- Plugin : extension qui, en s'ajoutant à la bibliothèque initiale, en étend les possibilités
- Mise en place très simple : télécharger le ficher correspondant et l'inclure dans le document HTML de la même façon que jQuery.
- Exemples : jqueryui.com/demos/
  - ensemble considérable de widgets : panneaux, menus, accordéons, onglets, arbres, fenêtres pop-up…
  - Validation de formulaires
  - Tables
  - Time, Date and Color Picker
  - Google Map
  - Drag and Drop
  - ....

Plugins jQuery très nombreux

- A tester en Tp si vous avez le temps
- Très pratiqué en stage dans des stages de développement web !

- Amusez-vous bien avec