Cours jQuery

Module: Technologie web 2.0

Préparé par: Up Web

Classe: 3ème année ESPRIT

Plan du cours

I. Introduction

II. Les sélecteurs

III. Les évènements

IV. Quelques effets

V. Manipulation du DOM

I. Introduction

1. Principe
2. Utilisation
   a. Chargement de la bibliothèque
   b. Lancement au chargement de la page
3. Utilisation de plugins

I. Introduction
1.Principe

- jQuery est une bibliothèque JavaScript qui a pour but de soulager le développeur des tâches fastidieuses de gestion de compatibilité inter-navigateurs, ainsi que de lui fournir des effets classiques « clef en main ».
- Une fonction incontournable de cette bibliothèque est la fonction jQuery() ou son alias **$()**, qui a de multiples usages comme nous allons le voir.
- Le but de ce chapitre n'est pas de donner une liste détaillée de toutes les propriétés et méthodes définies par cette bibliothèque. Le site officiel de Jquery http://jquery.com/ le fait bien plus complètement ; il s'agit simplement de fournir les bases permettant de saisir le principe de fonctionnement de la bibliothèque.

I. Introduction
2.Utilisation

a. Chargement de la bibliothèque

- Pour commencer, il faut évidemment télécharger la bibliothèque, qui est disponible sur le site officiel. Il suffit ensuite de l'incorporer à l'aide d'un élément script dans l'entête du document HTML :

```
<script type="text/javascript" src="CheminRelatifVersLeFichierjQuery.js"></script>
```

- Le script doit être présent sur le serveur afin de limiter les risques de rejet du code lié à la protection contre les scripts inter-domaines.

I. Introduction
2.Utilisation

b. Lancement au chargement de la page

- Pour ne lancer un script que lorsque l'on est sûr que l'intégralité du DOM a été chargée. jQuery offre une méthode plus souple, à l'aide de la méthode **ready** :

```
$(document).ready(GestionnaireALancer) ;
```

- On peut ainsi écrire :

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

- Ou bien :

```
$(document).ready(Gestionnaire) ;
function Gestionnaire(evt){...}
```

- En général une écriture jQuery suit le syntaxe suivant: **$(selecteur).action();**
- Il suffit de placer cette ligne de code entre les balises <script> et </script> dans l'entête du document HTML.

I. Introduction
3.Utilisation des plugins

- jQuery est très utile en combinaison avec des extensions, ou plugins. Il en existe de toutes sortes, disponibles sur le site officiel des plugins (http://plugins.jquery.com/), où des démonstrations sont possibles. Avant de réaliser un effet, il est souvent judicieux de vérifier s'il n'existe pas déjà un plugin qui permet de le réaliser, souvent avec des options attractives.

- Pour utiliser un plugin, il suffit de charger au préalable la bibliothèque, puis le fichier JavaScript de l'extension.

II. Les sélecteurs

1. Sélecteurs de base
2. Filtrage
    a. Filtre sur l'arborescence
    b. Filtre de contenu
    c. Filtre de visibilité
    d. Filtre d'attributs

II. Les sélecteurs
1. Les sélecteurs de base

- La fonction $ permet de sélectionner directement des éléments à l'aide de la syntaxe CSS . Du coup on distingue:

➔**$("#ident") ⇔ document.getElementById("ident")** en plus court

a] Sélecteur natif:

➔**$("h2")[1]** cible le deuxième élément h2 du document.

➔**$("p").length** donne le nombre de paragraphes dans un document.

➔**$("p")** sélectionne tous les éléments p du document.

b] Sélecteur de id:

➔**$(" #truc")** sélectionne l'élément portant l'identifiant truc.

c] Sélecteur de class:

➔**$(" .truc")** sélectionne les éléments portant la classe truc.

- **$("*")** sélectionne tous les éléments.
- Il est possible d'indiquer plusieurs sélecteurs, à la manière des CSS : **$(".truc, div, #machin")** sélectionne tous les éléments de classe truc, tous les éléments div et l'élément d'identifiant machin.

II. Les sélecteurs
2. Filtrage

a. Filtres sur l'arborescence

- **:first** sélectionne le premier élément d'une collection, **:last** le dernier. Par exemple, $(".truc:last") sélectionne le dernier élément de classe truc dans le document.

- **:not(selecteur)** permet de retirer de la sélection tous les éléments spécifiés. Par exemple, **$(".truc:not(.machin)")** permet de sélectionner tous les éléments de classe truc ne possédant pas la classe machin.

- **:header** sélectionne tous les titres (h1, h2, etc.)
- **:odd** et **:even** sélectionne tous les éléments d'ordre respectivement impair et pair d'une collection. Par exemple, $(tr:even) sélectionne les lignes de tableau n°0, 2, 4, etc.

II. Les sélecteurs
2. Filtrage

b. Filtres de contenu

- **:contains(texte)** sélectionne tous les éléments contenant un texte donné (par exemple:**$("p:contains('test')")** cible tous les paragraphes contenant le texte "test".

- **:has(sélecteur)** sélectionne les éléments contenant au moins un élément sélectionné par sélecteur. Par exemple, $("li:has(ul)") sélectionne les éléments d'item de liste (li) contenant au moins une liste.

- **:empty** sélectionne les éléments vides.

II. Les sélecteurs
2. Filtrage

c. Filtres de visibilité

- **:visible** sélectionne les éléments qui sont visibles
- **:hidden** sélectionne les éléments qui ont été cachés (voir ci-après)

d. Filtres d'attributs

- **[attribut]** sélectionne les éléments possédant l'attribut attribut.
- **[attribute=valeur]** sélectionne les éléments possédant un attribut *attribut* valant *valeur*. Par exemple:$("td[colspan=2]") sélectionne les cellules de tableaux s'étendant sur deux colonnes.

III. Les évènements

1. Evénements du DOM
2. Nouveaux événements
3. Gestionnaires d’évènements
4. Exercice: Premiers effets
5. Exercice: Sélecteurs

III. Les évènements
1. Evènements du DOM

- Les événements de la spécification du DOM sont baptisés simplement en enlevant le préfixe "on" de l'attribut HTML correspondant : on obtient alors le nom de la méthode à appliquer à l'élément sélectionné.
- Par exemple, $("p").click(function(){alert("salut!")})

III. Les évènements
2. Nouveaux Evénements

- jQuery définit de nouveaux événements. En voici quelques-uns :
- **mouseenter** est lancé quand la souris pénètre « au-dessus » d'un élément. Il n'est actif qu'à l'*entrée* de la souris, contrairement au **mouseover** qui, lui, est lancé aussi quand la souris survole l'élément. Il est associé à **mouseleave**, qui est actif quand la souris quitte l'élément.
- **scroll** est lancé quand l'utilisateur fait défiler la page.

III. Les évènements
3. Gestionnaires d'événements

- Pour mémoire, un gestionnaire d'événement est une fonction destinée à être lancée en réponse à une action de l'utilisateur (par exemple un clic de souris sur un élément donné). De la même manière qu'avec le DOM, jQuery fournit deux manières de définir un gestionnaire d'événement :
- soit en indiquant le *nom* du gestionnaire, par exemple...
- **$**("p").**click**(Gestionnaire) ;

```
function Gestionnaire(evt)
{
alert("Ceci est un paragraphe") ;
}
```

- soit en codant directement le gestionnaire, par exemple...
- **$**("p").**click**(function(){alert("Ceci est un paragraphe");}) ;

III. Les évènements
4. Exercice: Premiers effets

- Insérer deux paragraphes dans une page html
- Affecter aux deux paragraphes le gestionnaire clickP associé au clic. Cette fonction change la couleur de l'élément cliqué en rouge.

III. Les évènements
5. Exercice: Sélecteurs

- Au chargement de la page, affecter aux deux paragraphes de classe rouge le gestionnaire clickRouge associé au clic.
- Ce gestionnaire associe ensuite au clic sur chacun des paragraphes *qui ne sont pas* de classe rouge le gestionnaire clickP, qui affiche l'identifiant de l'élément cliqué dans une boîte d'alerte.

IV. Quelques effets

1. Apparition, disparition
2. Effets personnalisés, contrôle
3. Exercice: Effets

IV. Quelques effets
1. Apparition, disparition

- **show()** et **hide()** permettent respectivement de montrer et cacher des éléments. Par exemple, $("p").hide() cache tous les paragraphes du document.
- **show(vitesse)** et **hide(vitesse)** permettent respectivement de montrer et cacher des éléments avec une certaine vitesse. Cette vitesse est indiquée par des mots-clefs (**"slow"**,**"normal"** ou **"fast"**) ou le nombre de millisecondes que doit durer l'animation.
- **toggle()** et **toggle(vitesse)** permettent de basculer d'un mode d'affichage à un autre (un élément caché devient visible, ou un élément visible devient caché).
- **slideDown()** et **slideUp()** permettent de faire apparaître (respectivement disparaître) un élément à la manière d'un store se déroulant ou s'enroulant.
- **slideToggle()** permet de basculer d'un mode d'affichage à un autre.
- **fadeIn(vitesse)** et **fadeOut(vitesse)** permettent de faire progressivement apparaître (ou disparaître) un élément en jouant sur sa transparence.

IV. Quelques effets
2. Effets personnalisés, contrôle

- **animate(paramètres)** permet de contrôler une animation, via par exemple les propriétés CSS.
- **stop()** arrête toutes les animations en cours sur le document
- **jQuery.fx.off = true;** permet de désactiver toutes les animations d'un document.
- **jQuery.fx.off = false;** les réactive.

IV. Quelques effets
3. Exercice: Effets

- Au chargement de la page, associer les gestionnaires clickPair et clickImpair respectivement aux items de liste numérotés par des nombres respectivement pair et impair (attention, la numérotation commence à 0...).
- Le gestionnaire clickImpair permet d'afficher et de masquer l'autre liste. Par exemple, si on clique sur "Item 1.1", la seconde liste disparaît ; elle réapparaît cependant si on clique sur, par exemple, "Item 1.3".
- Le gestionnaire clickPair fait la même chose, mais avec un effet de déroulement.

V. Manipulation du DOM

1. Création des noeuds
2. Modification des noeuds
3. Insertion de contenu
   a- Insertion à l'intérieur d'un élément donnée
   b- Insertion à l'extérieur d'un élément donnée
   c-Insertion autour d'un élément donnée
4. Remplacement et suppression
5. Exercice: Manipulation du DOM

V. Manipulation du DOM
1. Création des noeuds

- La fonction $ permet de facilement créer des nœuds. par exemple: `$("<div><p>Un peu de texte</p></div>")`

➔ crée un élément div contenant un paragraphe.

- Attention cependant, un tel élément reste « en suspens » tant qu'il n'a pas été explicitement rattaché au DOM, de même qu'un élement créé avec la méthode createElement().

V. Manipulation du DOM
2. Modification des noeuds

- jQuery facilite l'accès et la modification des contenus des nœuds.
- Le code HTML est accessible via la méthode html(). Si cette méthode est appelée sans argument, elle renvoie le contenu HTML de l'élément sélectionné. Ainsi, si on a:

```
<p id="p1">Un peu de texte<em>important</em>.</p>
```

- **alert($("#p1").html());**

➔affichera **Un peu de texte important**

- Si cette méthode est appelée avec argument, alors elle permet de modifier le contenu HTML. Toujours avec le même code source HTML,
- **alert($("#p1").html("Un peu de texte très important."))**

➔ a pour effet de remplacer le code source initial.

- La méthode text() permet, elle, de lire ou de modifier le contenu textuel des éléments auxquels elle est appliquée. **$("#id1").text()** lit ainsi le contenu de l'élément d'identifiant **id1**, alors que **$("li").text("salut");** permet de modifier le contenu de tous les éléments li en "salut".

V. Manipulation du DOM
3. Insertion de contenu

a. Insertion à l'intérieur d'un élément donné

- Les méthodes **append(contenu)** et **prepend(contenu)**

ajoutent contenu à l'élément sélectionné, respectivement à sa fin et à son début.

- Les méthodes **appendTo(sélecteur)** et **prependTo(sélecteur)**

ajoutent l'élément sélectionné à la fin (respectivement au début) de l'élément spécifié par sélecteur. Par exemple:

```
$("<span>(Fin de paragraphe)</span>").appendTo($("p"));
```

➔ Ajoute un élément span à la fin de tous les paragraphes du document.

V. Manipulation du DOM
3. Insertion de contenu

b. Insertion à l'extérieur d'un élément donné

- **after(contenu)** et **before(contenu)** sont des méthodes qui ajoutent contenu respectivement après et avant l'élément sélectionné.
- **insertAfter(sélecteur)** et **insertBefore(sélecteur)**

  sont deux méthodes qui ajoutent l'élément sélectionné après (respectivement avant) l'élément spécifié par sélecteur. Par exemple:

```
$("<p>(Fin de section)</p>").insertBefore($("*:header:not(h1)"));
```

➔Ajoute un paragraphe avant tous les titres du document, à l'exception des titres h1.

V. Manipulation du DOM
3. Insertion de contenu

c. Insertion autour d'un élément donné

- **wrap(html)** permet d'intégrer l'élément sélectionné dans le code HTML spécifié.
- **wrap(élément)** permet d'intégrer l'élément sélectionné dans l'élément spécifié.
- Par exemple:

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

➔Permettent d'entourer tous les paragraphes par un élément div.

V. Manipulation du DOM
4. Remplacement et bsuppression(1)

- **replaceAll(sélecteur)** permet de remplacer l'élément indiqué par sélecteur par le contenu spécifié, exemple:

```
$("<p>Paragraphe</p>").replaceAll("h3");
```

➔Remplace tous les titres de niveau 3 par des paragraphes avec le même contenu.

- **replaceWith(contenu)** permet de remplacer l'élément sélectionné par contenu :

```
$("em").click(function(){$(this).replaceWith("
<strong>"+$(this).text()+"</strong>")});
```

➔remplace ainsi au clic un élément em par un élément strong.

V. Manipulation du DOM
4. Remplacement et bsuppression(2)

- Pour vider un élément sélectionné, on fait appel à la méthode **empty()**. Par exemple: `$("#p1").empty()`

➔laisse présent l'élément d'identifiant p1, mais supprime tous ses enfants.

- La méthode **remove(expression)** supprime de l'élément sélectionné le contenu indiqué par expression Par exemple:

```
$("p").remove($(":contains('test')"));
```

➔supprime tous les paragraphes contenant la chaîne de caractères "test".

V. Manipulation du DOM
5. Exercice: Manipulation du DOM

- Au chargement de la page, affecter les gestionnaires suivants au clic sur les listes :
- Le gestionnaire ajoute associé à la première liste lui ajoute un élément d'item de liste (li). Cet item contient le texte "Item" suivi du numéro de l'item (par exemple, au premier clic, on obtient une liste dont le troisième item contient "Item 3").
- Le gestionnaire retire associé à la deuxième liste enlève le dernier item de la première liste.
- Le gestionnaire remplace associé à la troisième liste remplace son premier item par le dernier de la première liste. Que se passe-t-il quand on utilise replaceWith ?

Question?