IUT – Département informatique
2ème Année – Formation initiale

février 2016

Jean-Michel Ilié, Camille Kurtz

TP
connexion dynamique en jquery

But du TP : entraînement à JQUERY et AJAX

Vous avez à votre disposition un canevas d'application PHP-MVC,
pour laquelle nous ne nous intéressons qu'à la connexion d'un utilisateur référencé.
Initialement, la page HTML du site s'affiche et son entête contient un bouton `connexion`.
Il ne s'agit pas de contrôler son accès mais de reconnaître l'utilisateur.

Le squelette de l'application MVC est le suivant

- index.php
- C
  - user.php
- V
  - page.tpl
  - connect.tpl
  - css
    - connect.css
  - js
    - connect.js
- M
  - user_bd.php

PARTIE 1 : Affichage du formulaire de connexion

Il existe un formulaire dans l'entête de la page HTML. Initialement, celui-ci n'est pas visible du fait de la propriété de style `display:none`. Développer la gestion événementielle consistant à permettre de cliquer sur le bouton `connexion`. Alternativement, le formulaire doit apparaître et disparaître, ce qui revient à changer la valeur de display de `none` à `block`, et réciproquement.

- Attacher à votre fichier HTML, le fichier `V/js/connect.js` spécifiant la gestion événementielle de cette application. Ce fichier est déjà créé.
- En JQUERY, développer votre code de déclaration de gestion événementielle (`click`) dans une fonction passée en paramètre d'un objet JQUERY. Elle sera exécutée juste après le chargement de la page HTML en mémoire:

```
$(
```

function(){...faire ici les déclarations des gestions événementielle ....}
);

- Programmer en JQUERY, par exemple, le code suivant créée une variable `v` initialisée. Le tableau interne de l'objet jquery créé (à savoir, le $) est affecté avec la référence d'un objet-balise d'identifiant `id='d'`, et `v` est affectée avec la valeur de l'attribut `href` de la balise correspondante.

```
var v=$('#d').attr(href);
```

- Les déclarations de click en jquery, se font de différentes manières, par exemple, en utilisant la fonction `click()` de l'objet `$`. Son paramètre est une référence à la fonction à lancer en cas de clic, sur l'un des balises référençées dans le tableau interne de l'objet `$`.

```
$.click(
      function(){… spécifier ici le code de réaction au clic…}
);
```

- *Voir le cours sur jquery pour plus d'informations sur les possibilités offertes par l'objet jquery*.
    - Lancer l'affichage initial de la page du site. Pour cela, la fonction `page()` du fichier de contrôle `C/user.php` est déjà écrite pour produire cette page, à partir de l'url :

```
index.php
```

PARTIE 2 : Soumettre le formulaire en AJAX

L'attribut action du formulaire est déjà initialisé pour qu'un clic sur le bouton de soumission invoque le script PHP `index.php`, suivant un format MVC :

```
index.php?control=user&action=connect.
```

- La fonction `connect()` du fichier de contrôle `C/user.php` est déjà écrite pour reproduire la page du site, mais complétée de la valeur du `login`, placé dans l'entête sous le bouton de `connexion`, au sein de balise `<h5></h5>`.
- Dans cette version de l'application, la vérification en base de données, du login et mot de passe saisis, est intentionnellement fictive, pour simplifier. En effet, la fonction Modèle `connect_bd()` de `M/user_bd.php` renvoie systématiquement true. Vous pourrez changer la valeur de ce return ou implémenter l'accès à la base ultérieurement.
- Des variables PHP sont exploitées dans le fichier template HTML, `V/connect.tpl`, de façon à afficher la page HTML avec ou sans spécification de login connecté.

De façon, à ne pas avoir toute la page à charger, gérer la soumission en javascript de façon à faire une requête AJAX vers un script PHP faisant juste la vérification (ici fictive) et renvoyant la valeur du login. Il vous restera à placer la valeur reçue dans l'entête de la page existante sur le navigateur.

- Ajouter au fichier javascript, la gestion directe de la soumission du formulaire, en utilisant la fonction `submit()` d'un objet `$`. Le tableau interne

de cet objet devra être affecté avec la référence de l'objet-balise correspondant au formulaire.

```
$.click(
      function(){… spécifier le code de réaction au submit…}
);
```

- Le code de réaction à une soumission consiste principalement à invoquer la fonction `ajax()` de l'objet `$`. L'objet passé en paramètre précise la requête. *Voir un exemple plus loin.*

```
$.ajax(
      {propriétés d'un objet json}
);
```

Les propriétés de l'objet passé en paramètres : doivent préciser
(1) Propriété `type` : la méthode de transmission POST,
(2) Propriété `data` : la chaîne sérialisée des données du formulaire, récupérée avec la fonction `$.serialize()` si le tableau interne de l'objet `$` contient la référence du formulaire.
(3) Propriété `url` : l'url du script PHP à invoquer sachant que l'action PHP à lancer s'appelle `ajax_connect()`

```
index.php?control=user&action=ajax_connect
```

Noter d'emblée que l'action invoquée en PHP est située dans le fichier de contrôle `C/user.php` et a pour nom `ajax_connect()`.
(4) Propriété `success` : référence d'une fonction consistant à placer la valeur du login reçue dans l'entête sous le formulaire.
(5) Propriété `error()`: référence d'une fonction consistant à afficher un message d'erreur et une invite à recommencer la soumission, en cas d'erreur sur la transmission ajax.

- Enfin, il faudra éviter le lancement de la soumission par défaut du formulaire (avec l'url indiquée dans le paramètre action du formulaire), en terminant la fonction de réaction par :

```
return false;
```

Noter que dans ce cas, la précédente action `connect()` de `user.php` ne sera plus d'utilité.

<u>Exemple d'utilisation de la fonction $.ajax :</u>

```
$.ajax({
    url: 'url_adressant_un_script_php',
    type: 'POST',       /*ou GET*/
    data: {/* paramètres ou données transmis au script php*/
        login:'jmi', /*c'est un exemple*/
        pass:19
    },
   error: function() {
     // En cas d'erreur, on le signale en écrivant cette fonction
        },
   success: function(data) {
     // ici, data est la donnée transmise par le script PHP
       // et on écrit la fonction success pour l'exploiter
        }
  dataType : html  /*Cas où la donnée retournée est un texte.
                     Autres valeurs pour dataType: 'json', 'xml'.*/
});
```

- Implémenter en PHP la fonction `ajax_connect()` du fichier de contrôle `C/user.php`. Dans cette fonction, lancer la fonction Modele `connect_BD()` de `M/user_bd.php` pour en tester le résultat : En cas de résultat de positif du test, faire un `echo` de la chaîne `'OK'` et en cas contraire renvoyer `'KO'`.
- Reprendre le code la fonction javascript, et implémenter la fonction de la propriété success suivant 2 cas : si la donnée reçue du fait de la requête ajax est `data='OK'` alors recopier la valeur du login sous le formulaire (balise <h5>), sinon préciser à ce niveau à l'utilisateur de recommencer sa demande de connexion.