

Programmation Web

Ajax et Web 2.0

David Odin

31 mai et 1^{er}, 8 et 15 juin 2007
TP sur deux séances

1 Présentation

Ce TP propose d'utiliser les technologies « « Web 2.0 » » vues en cours afin de réaliser un site le plus possible agréable à utiliser.

Les premières parties vous permettront de vous familiariser avec les différents concepts, objets et fonctionnalités des bibliothèques *prototype* et *scriptaculous*.

La dernière partie sera consacrée à la réalisation d'un site web, côté serveur (PHP) et côté client (Javascript), qui utilisera les facilités offertes par l'API de Flickr.

2 Prototype

Vous trouverez à partir de la page <http://prototypejs.org/api/> toute la documentation nécessaire pour utiliser la bibliothèque `prototype.js`.

Afin de l'utiliser simplement, créez un sous-répertoire **bin** et placez-y le fichier **prototype.js** que vous trouverez à partir de la page <http://prototypejs.org/download>

Ensuite dans un fichier *index.html* ou *index.php*, vous pourrez l'utiliser de cette façon :

```
<head>
...
<script type="text/javascript" src="bin/prototype.js"> </script>
<script type="text/javascript" src="bin/monTP.js"> </script>
...
</head>
```

Il ne vous restera plus qu'à écrire le fichier *bin/monTP.js*.

2.1 Utilisation de \$ et de l'objet Event

À partir de la documentation de Prototype, réalisez une page possédant au moins deux éléments, peu importe leur type. Lorsque l'utilisateur cliquera sur le premier élément, le second devra être caché. Et si l'utilisateur clique à nouveau sur le premier élément, le second devra être réaffiché.

La sélection des éléments se fera par l'intermédiaire des fonctions `$()` et `$$()`.

2.2 Les insertions

Utilisez l'objet *Insertion* de *prototype* pour ajouter des éléments à votre page après son chargement.

2.3 Ajax

Pour cette question, il vous faudra également créer une deuxième page (un script php).

Utilisez la fonction `Request` de l'objet `Ajax` de *prototype* pour demander l'heure au serveur (par le biais d'un script php).

L'utilisateur devra presser un bouton sur la page principale, et les différentes heures devront s'afficher les unes derrière les autres dans la page principale (sans la recharger...)

Vous pouvez également utiliser `Ajax.Updater` pour cela.

3 Scriptaculous

Vous pouvez télécharger la bibliothèque scriptaculous à partir de <http://script.aculo.us/downloads>

Après décompression, placez les différents fichiers `.js` présents dans le répertoire `src` dans votre répertoire `bin`.

Comme Scriptaculous est une surcouche de prototype, il faut inclure `prototype.js` avant `scriptaculous.js`, par exemple de cette manière :

```
<head>
...
<script type="text/javascript" src="bin/prototype.js"> </script>
<script type="text/javascript" src="bin/scriptaculous.js"> </script>
<script type="text/javascript" src="bin/monTP.js"> </script>
</head>
```

3.1 Effets

À partir de la documentation donnée ici : <http://wiki.script.aculo.us/scriptaculous/tags/effects>, vous créerez une page mettant en œuvre un maximum d'effets utilisant l'objet **Effect** de scriptaculous.

3.2 Glisser-Déplacer

À l'aide de la documentation présente ici : <http://wiki.script.aculo.us/scriptaculous/show/DragAndDrop>, vous devrez créer une page permettant à l'utilisateur de choisir les éléments qu'il préfère dans une liste donnée (par exemple une liste de chansons). Le choix se fera par un simple glisser-déplacer depuis la liste de toutes les chansons vers un élément comportant l'ensemble des morceaux choisis.

3.3 Complètement automatique

En partant des exemples donnés sur cette page : <http://wiki.script.aculo.us/scriptaculous/show/Ajax.Autocompleter>, vous devrez réaliser une page demandant à un utilisateur d'entrer un nom de département français.

L'utilisateur sera aidé dans son entrée : dès qu'il commencera à écrire le nom (ou le numéro) d'un département, l'entrée proposera la liste des départements correspondants.

La liste des départements est facile à trouver sur wikipedia ou quid.fr.

4 Utilisation de l'API de Flickr

Pour cette dernière partie, vous devrez réaliser un site un peu plus conséquent, en utilisant bien entendu tout ce qui vient d'être fait. L'ensemble de l'api flickr utilisable est disponible et documenté ici : <http://www.flickr.com/services/api/>

Parmi les fonctions présentées, un certain nombre ne nécessitent même pas d'être membre du site Flickr pour être utilisées :

- flickr.contacts.getPublicList
- flickr.favorites.getPublicList
- flickr.groups.getInfo
- flickr.groups.search
- flickr.groups.pools.getContext
- flickr.groups.pools.getPhotos
- flickr.interestingness.getList
- flickr.people.* sauf flickr.people.getUploadStatus
- beaucoup de flickr.photos
- et beaucoup d'autres

4.1 Utilisation

Cependant, il est nécessaire d'obtenir une clef pour utiliser ces fonctions, cela se fait simplement à partir de la page <http://www.flickr.com/services/api/keys/>

Plusieurs fonctions de l'api flickr renvoie des photos sous la forme d'un élément (XML ou autre), pour récupérer l'image correspondante, vous consulerez la documentation disponible ici :

<http://www.flickr.com/services/api/misc.urls.html>

L'api Flick est disponible via SOAP ou REST (je vous conseille vivement REST !)

Les réponses peuvent être dans différents formats, dont le format JSON (<http://www.json.org/js.html>) qui est particulièrement adapté à un développement en Javascript.

4.2 Travail demandé

À l'aide de toutes ces informations, vous proposerez une interface permettant à un utilisateur de visualiser toutes les photos correspondant à certains critères de recherche. Vous êtes libre de créer l'interface qui vous semble la plus agréable à utiliser.