Jonas Bouscharain
Tanguy Helbert
Groupe 2

INSTITUT UNIVERSITAIRE DE TECHNOLOGIE

Département INFORMATIQUE

(Computer Science Department – Nantes Institute of Technology)

Programmation Web Client Riche

2014-2015

Mini-projet

Client riche - jQuery

2ème année DUT Informatique

Réalisé par : Jonas BOUSCHARAIN
Tanguy HELBERT

Encadré par : Christine JACQUIN

06/04/2015

1. Introduction

Le projet, que nous avons eu à réaliser, consistait à construire une page web permettant à un utilisateur lambda de rechercher des photos relatives à des villes françaises. L'utilisateur aura la possibilité de visionner les photos de différentes manières et pourra les trier.

2. Développement

2.1 L'auto-complétion

Nous allons commencer par présenter l'auto-complétion. Puisque le script fournis n'était pas accessible depuis chez nous, nous avons utilisé une application de jquery ui nommée geocode. Celle-ci fonctionne exactement comme celui qui était fournis à la base mais est réalisé pour de l'international. Nous avons tenté de réduire les résultats pour la France, sans réussite.

2.2 La date minimale

La date minimale fait appel à une méthode de jquery ui, permettant l'affichage d'un calendrier et la sélection à la souris de cette date. Cette valeur va être formatée pour être utilisée ensuite dans la comparaison avec les autres dates. Nous avons créés des chiffres correspondant à une date, 2014-06-04 donnant 20140604, on peut donc comparer deux entiers pour vérifier si la date correspond.

2.3 Affichage d'images

Pour changer d'affichage, nous avons créés deux boutons « LISTE » et « CARROUSEL » qui déclenchent chacun une fonction consistant à afficher le format indiqué sur le bouton et cacher l'autre.

2.3.1 Collecte d'images

Avant d'afficher les images, un utilisateur peut sélectionner le nombre de photos qu'il souhaite afficher (5, 10, 20). Cette donnée sera utilisée dans la fonction getJSON.

Dans cette fonction nous allons fournir l'adresse du site flickr, précisant le flux public de photos publiées, ainsi que la valeur entrée dans l'input de recherche. Lorsque ces données sont récupérées, nous allons d'abord vider les affichages de l'ancienne recherche puis remplir nos affichages avec nos nouvelles données (liste et carrousel). Ce remplissage s'effectue en fonction de la valeur sélectionnée dans la liste déroulante (nombre maximale d'images) ainsi que la date choisie (si la date de la photo est plus ancienne que la date sélectionnée, celle-ci n'est pas affichée).

2.3.2 Carrousel

Pour afficher le résultat de la recherche sous forme de carrousel nous avons utilisé un plugin jquery nommé fotorama très simple d'utilisation. Il suffit d'importer leur script ainsi que leur css et crée une div , puis inséré les images à l'intérieur de cette div. Il est tellement simple d'utilisation qu'il crée des soucis à l'utilisation. En effet, après l'avoir initialisé, le plugin détecte le carrousel à créer et change directement notre code pour générer son application. Or, lorsque la recherche change, il devient difficile de changer le contenu de ce carrousel avec nos

nouvelles images. Nous avons tenté de multiples manipulations, comme vider tout le contenu et régénérer le code de base pour recréer un carrousel, malheureusement il ne se met pas à jour. Pour le moment, notre carrousel s'initialise à la première recherche et conserve, quand il le veut bien, la totalité des images de nos recherches.

2.3.3 Liste

Notre liste d'images est une liste div composé d'éléments possédants le style 'inline-block', ils vont donc s'ajouter les uns à côté des autres jusqu'à arriver au bord de la page. Ces images sont insérées avec une syntaxe bien spécifique puisqu'elles sont utilisées avec deux plugins. Le premier, magnific-popup, va permettre à l'utilisateur d'avoir une meilleure vue de la photo ainsi que quelques informations sur la photo (titre, date, auteur), une popin va s'affiché au click sur une image. Toutes ces informations sont gérées par le plugin et doivent être insérées dans le titre de notre image. Ces informations sont encapsulées dans une autre div servant au tri des éléments. Ce tri est effectué par le plugin mixItUp qui va comparer les éléments en fonction d'attributs personnalisés de nos images. Par exemple, nous avons rajouté l'attribut 'data-date' à chaque image pour permettre au plugin, au clic sur un bouton configuré, de comparé les images par rapport à cet attribut et les triées dans l'ordre que l'on veut (ascendant, descendant, aléatoire, etc.).

2.3 Fenêtre modale

Nous avons défini une fenêtre modale qui s'affiche lorsqu'aucune image n'est détectée pour la recherche d'une ville. Pour cela, nous avons défini une variable, dans notre fonction de collection d'images, initialisée à 0. Celle-ci s'incrémente lorsque la méthode trouve une image. Si, à la fin de la fonction la variable est toujours nulle la fenêtre est affichée à l'aide des méthodes show. Pour l'afficher nous avons utilisé le css utilisé lors du TD1 de jquery.

3. Conclusion

Au final, ce projet a été intéressant et enrichissant pour nous car il nous a permis de mettre en pratique ce que nous avons appris au cours des séances de TD de façon plus poussée ainsi que de découvrir le nombre incroyable de travaux disponibles sur le web utilisant jquery.