Université de Caen Basse Normandie
Système - François Rioult

Devoir NodeJs : développer un mini wiki

Cyril JIMENEZ et Alizée TIERCERY

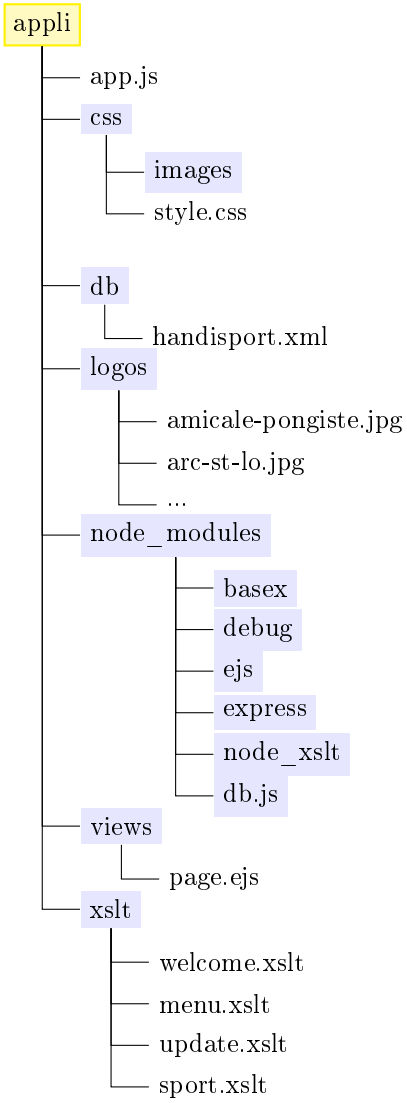
14 janvier 2014

Sommaire

1 Introduction

Il s'agit de développer une application en node js qui se comporterait comme un wiki. On pourra ajouter, modifier ou supprimer des pages. On traite des donnes XML qui agissent comme base de données. Nous avons choisi d'utiliser comme sujet pour remplir le wiki, les clubs sportifs adaptés à Caen pour accueillir les personnes handicapées.

2 Architecture



Nous avons un script js principal 'app.js' qui lance le serveur web. Le dossier **node_modules** contient tous les modules nécessaires pour l'application. Nous avons aussi créé notre propre module « db.js » qui s'occupe de faire des requêtes sur le fichier XML. Dans le dossier **xslt**, on y met les fichiers xslt pour l'affichage, qui permettent de transformer le xml en html. Dans **views**, les deux scripts d'ajout et de modification. Un dossier **CSS** pour le style, et un dossier **db** qui contient le fichier XML qui nous sert de base de données.

3 Modules

Nous avons téléchargés plusieurs modules nécessaires pour notre site :

- **Express** : framework node pour gérer le routing notamment.
- **Debug** : permet d'afficher des objets de façon détaillée (équivalent d'un var_dump en PHP).
- **Node_xslt** : permet de transformer le XML issu des requêtes en BDD directement en HTML.
- **Ejs** : moteur de template pour node. Permet de faire des inclusions dynamiques, ce qui semble impossible en XSLT.
- **Basex** : connexion à la BDD et requêtes Xquery.

4 Constitution du fichier XML « handisport.xml »

Voilà comment est formé notre fichier xml :

```
<sports>
  <sport>
<title>Cyclisme</title>
<slug>cyclisme</slug>
<description>A la fin... </description>
<links>
  <link>http://fr.wikipedia.org/wiki/Cyclisme_handisport</link>
</links>
<clubs>
  <club>
<name>Amicale cycliste de Bayeux</name>
<address>Place de la gare - 14400 BAYEUX</address>
<phone>06.74.64.43.03</phone>
<mail>michel.david11@orange.fr</mail>
<website>www.amicale-cycliste-bayeux.fr</website>
  </club>
  <club>
<name>Sport Handi Nature</name>
<address>135 rue de Bayeux - 14000 CAEN</address>
<hours>Loisir : Le samedi après-midi de 13h30 à 17h30
Horaires Compétition En semaine suivant les
  </hours>
<phone>06.74.66.34.50</phone>
<mail>gaby.bourgault@gmail.com</mail>
<website>www.sporthandinature.fr</website>
  </club>
</clubs>
  </sport>

  <sport>
```

```
 ... ... ... ... ...
  </sport>
</sports>
```

5 Fichier app.js

Ce script joue le rôle de contrôleur : il intercepte les requêtes http, interroge la base de données et appelle la vue correspondante (généralement sous la forme d'un fichier XSLT). Il crée le HTML résultant de la transformation du XML, et le met dans une variable qui sera transmise au squelette de la page en EJS.
Le code simplifié :

```
// importation des modules
...

// routage
app.get('/mon-url', function(req, res) {
        db.requete(params, function(réponse en XML) {
                var content = transform('nom-de-la-feuille-xslt', réponse en XML) ;
                // content contient le HTML
                end('titre de la page', content, res) ;
        }) ;
}) ;

// transforme le XML passé en paramètre à l'aide de la feuille XSLT
// passée en paramètre
function transform('nom', xml) {
...
}

// envoie la réponse finale au client
function end(titre de la page, content, res) {
        // récupère la liste des sports pour écrire le menu, présent sur
        // toutes les pages
        db.getList(function(list) {
                var menu = transform('menu.xslt', list) ;
                // nous renvoyons toujours le même squelette de page,
                // seul le contenu change
                res.render('page.ejs', { titre de la page, content, menu}) ;
        }) ;
}
```

6 Fichier db.js

Fonctions principales, requêtes XQuery :

- **getList()** : récupère la liste des sports (le titre)
- **getSport()** : récupère les données du sport sélectionné
- **deleteSport()** : supprime le sport sélectionné (le supprime dans basex et dans le fichier xml)
- **addSport()** : rajoute en base et dans le fichier XML un sport. Notre formulaire d'ajout est simple, nous n'avons pas géré la possibilité d'entrer plusieurs clubs ou plusieurs url pour un même sport.

Exemple de code :

```
var getList = function(callback) {
        var input = "<liste>{ requête Xquery } </liste>";
        var query = basex.query(input);

        query.results(function(err, reply) {
                callback(reply.result);
        });
}
```

7 Difficultés rencontrées

Une des plus grandes difficultés que nous avons eues est que les requêtes à la BDD sont faites de façon asynchrone. Concrètement, si on essaye de mettre un « return reply.result » à la place du « callback(reply.result) », cela ne fonctionne pas car la fonction anonyme passée en paramètre de query.results est exécutée seulement à la réception du résultat de la requête, soit quelques millisecondes plus tard. Pendant ce temps, l'exécution de la fonction getList se poursuit, et arrive à la fin sans rien renvoyer. Le return que l'on aurait mis ne servirait à rien car d'une part il arriverait trop tard, et d'autre part il est encapsulé dans une fonction anonyme donc impossible à récupérer.
Pour résoudre ce problème, nous utilisons le principe des callbacks, très utilisés dans node. Il s'agit simplement de passer en paramètre à getList une fonction, qui sera exécutée à la réception des résultats.

Autre problème : lors d'une requête qui met à jour la BDD (insertion, suppression, modification), le fichier XML d'origine n'est pas modifié, seule la représentation de la base dans la mémoire du serveur basex l'est. Il fallait simplement utiliser la fonction export après chaque requête pour corriger ce problème, mais nous avons mis du temps pour trouver la solution.

8 Fonctionnalités supplémentaires

Nous avons également réalisé un upload d'image avec node, pour les logos des clubs. Enfin, nous avons utilisé un module pour générer automatiquement les slugs (les identifiants uniques de chaque sport que l'on peut transmettre dans l'url) à partir du nom du sport.