Pronote : Gérer les connexions à MongoDB avec votre serveur

Présentation

On a créé un serveur, sur lequel on a installé MongoDB, qui nous permet de sauvegarder des données. On va maintenant voir comment les récupérer, les supprimer, ou en ajouter directement avec un script javascript.

On a déjà un fichier serveur.js, qui contient les paramètres de notre serveur. C'est ce script qui contiendra les différentes fonctions pour ajouter, supprimer ou consulter des données.

Connexion à MongoDB

On va commencer par inclure le module qui gère MongoDB, avec la ligne de code suivante :

```
 var MongoClient = require('mongodb').MongoClient
  , assert = require('assert');
```

Il nous faut maintenant nous connecter sur notre base de données pour accéder à son contenu. On va créer une variable qui contiendra le lien de notre base de données :

```
var urlmdb = 'mongodb://localhost:27017/resultats';
```

“resultats” est le nom de la base de données qui contient ce sur quoi on veut agir.

Récupération des données sur MongoDB

Maintenant qu'on est connectés à MongoDB,on va pouvoir commencer à agir sur les données qu'il contient. On va donc créer une fonction qui permettra de récupérer les données.

Cette fonction lancera la connexion à MongoDB, puis ira chercher dans le dossier précisé les données voulues. Voilà le code de la fonction, commenté :

```
server.get('/get/', recupData); // Ajout de l'écoute sur le lien
10.187.37.160:8080/get/

function recupData(req, res, next){
    MongoClient.connect(urlmdb, function(err, db) { // Connexion à mongoDB
        assert.equal(null, err);
        console.log("GET : connexion a mongodb effectuee");

        var collection = db.collection('resultats'); //On précise le dossier
dans lequel les données sont stockées
```

```
        filtre = {"eleve": "DUPONT"}; // On précise les données qu'on veut. Ici, on va récupérer toutes les données contenant "eleve":"DUPONT"

        collection.find(filtre).toArray(function(err, resultat) { // La fonction va rechercher les données correspondantes à notre demande
            assert.equal(err, null);
            db.close();            // On referme la connexion à MongoDB
            resultat = JSON.stringify(resultat);    // On modifie le format du résultat renvoyé par la fonction
            res.end(resultat);            // On affiche ce résultat sur l'écran du client
            console.log(resultat);        // On affiche ce résultat dans la console
            });
    });
next();
}
```

Avant de tester ce code, pensez à créer, dans MongoDB, une ligne qui contient “eleve”:“DUPONT”. Pour tester ce code, il suffit de lancer le serveur, comme fait précédemment, et de mettre dans votre navigateur l'URL 10.187.37.160:8080/get

Pour l'instant, on ne récupère pas les données qui concernent la personne que le client souhaite voir, mais seulement celles de DUPONT.

Ce n'est pas si compliqué, on l'a fait un peu plus tôt, il suffit de mettre un écouteur sur l'URL saisi pour récupérer le nom, que l'on stocke dans une variable, laquelle sera à la place de DUPONT lorsqu'on définit le filtre.