Module Nouvelles Technologies du Web

TME - Développement avec Play! Framework

I°) Objectifs

L'application à réaliser sera un petit client BitTorrent où chaque étudiant pourra partager ses fichiers et récupérer ceux de ses camarades.

II°) Installation

Télécharger la version 2.3 de Play! sur le site. Respectez les étapes d'installation vu en cours.

III°) Modèle

Créer une classe TorrentFile, qui représentera un fichier distant dont nous avons connaissance. Ces fichiers ne seront pas disponibles localement sur les machines et correspondront aux fichiers connus par vos voisins (donc les autres étudiants).

Un TorrentFile est une classe sérializable content un nom de fichier, une adresse ip et un port. Les deux derniers champs représentent donc l'adresse ip et le port du serveur sur lequel se trouve le fichier.

IV°) Service

Créer l'interface TorrentService. Les méthodes à définir sont les suivantes :

- List<String> getLocalFiles() → Retourne la liste des fichiers définis dans le répertoire ./files ;
- File getLocalFile(String name) → Retourne le fichier spécifié, disponible localement pour un téléchargement ;
- boolean hasLocalFile(String name) → Retourne vrai si le fichier spécifié est disponible au téléchargement, donc disponible localement sur la machine ;
- void index(String ip, int port, String name) → Ajoute un fichier distant (un TorrentFile) ;

- TorrentFile getRemoteFile(String name) → Retourne le TorrentFile associé au fichier spécifié.

Une fois l'interface créée, créer une classe TorrentIOService implémentant TorrentService.

TorrentIOService est un service permettant d'indexer les fichiers distants et de renvoyer les fichiers locaux. Cette classe possède une Map associant un nom de fichier à un TorrentFile (la liste des fichiers distants que nous avons indexé, vide par défaut) et un path, représentant le nom du répertoire dans lequel se trouve les fichiers locaux accessibles au téléchargement (par exemple ~/Vidéos). Cette valeur est mise par défaut à la création de l'instance.

Implémenter toutes les méthodes définies dans l'interface.

Faites valider votre travail par l'encadrant.

V°) Controller

Créer une classe Application héritant de Controller. Cette classe sera l'interface offrant des services web pour les utilisateurs distants. Elle contient 3 méthodes :

- Result download(String name) → Permet à un client de télécharger un fichier. Si le fichier est disponible sur le disque, il est renvoyé dans une requête 200. S'il est disponible à distance, une requête redirect est renvoyé, contenant le TorrentFile associé au format JSON. Enfin, s'il n'existe pas, une requête 404 est renvoyé, avec le nom du fichier.
- Result getFiles() → Renvoie au format JSON l'ensemble des fichiers disponibles localement.
- Result index(String ip, int port) → Effectue une requête HTTP sur l'ip / port spécifié pour obtenir la liste des fichiers disponibles chez le client spécifié. Cette requête a pour vocation d'appeler la méthode "Result getFiles()" chez un client distant. Une fois la requête effectuée, utiliser la fonctionnalité "map" et les "Promise" pour parcourir le JSON renvoyé et indexer chaque fichier donné. Cette requête renvoie toujours un code 200.

Enfin, dans le fichier routes, ajouter le mapping des URL que vous venez de créer. Toutes les métyhodes auront la méthode HTTP GET (pour faciliter les tests via le navigateur) et comme mapping respectif "/download", "/files", "/index". Le mapping "/index" aura 9000 comme valeur de port par défaut.

Faites valider votre travail par l'encadrant.

VI°) Configuration

Afin de rendre l'application plus modulaire, nous allons mettre le répertoire dans lequel les fichiers sont téléchargés dans la configuration de l'application. Ajouter une propriété "app.download.folder" dans "application.conf" et faites en sorte que TorrentIOService utilise cette valeur lorsqu'il est créé.

Faites valider votre travail par l'encadrant.