TP TRAITEMENT DE DONNEES OPEN FOOD FACTS

Objectifs

Vous connaissez peut-être l'application **Yuka,** disponible sur smartphone. **Yuka** fournit des informations nutritionnelles sur pratiquement tous les produits alimentaires commercialisés en France. En plus d'informations, elle fournit également un score nutritionnel, de A (excellent) à F (mauvais).

Cette application à succès s'est construite sur une **base de données open source** appelée **Open Food Facts**.

La base de données **Open Food Facts** est une base de données **mondiale**, qu'on peut télécharger sous la forme d'un fichier CSV. Le fichier que je vais vous demander de traiter dans le cadre de ce TP est le même que celui sur lequel s'est basé Yuka. Il ne concerne que **les produits alimentaires fabriqués en France**.

Dans ce TP vous allez créer une application fournissant des services pour rechercher des informations dans ce fichier. Dans un TP ultérieur, nous mettrons ce fichier en base de données, mais à ce stade les informations contenues dans le fichier seront traitées et chargées en mémoire au démarrage de l'application.

A savoir

- Les opérations de traitements de fichiers sont des opérations courantes en entreprise et il n'est pas rare d'avoir à en développer pour une application donnée.

- Il existe un nombre incalculable de données open source dont vous pouvez faire l'exploitation commerciale : données financières, échanges commerciaux, données sociales, agricoles, météorologiques, données satellitaires, etc... Si vous avez une idée, foncez !

Description du contenu du fichier

Le fichier fourni, au format CSV, comporte 13 432 références de produits avec 30 informations associées par produit :

- catégorie
- marque
- nom
- score nutritionnel : A (excellent) à F (mauvais)
- liste des ingrédients
- énergie pour 100g (en joules)
- quantité de graisse pour 100g
- liste des allergènes
- liste des additifs
- présence d'huile de palme
- toutes les vitamines,
- calcium, magnesium, etc.

Lors du cours Java JDBC nous mettrons ces données dans une base de données.

Type de fichier

Le fichier fourni est un fichier CSV :

- la première ligne est la ligne d'entête qui fournit le nom des attributs séparés par le caractère pipe « | ».
- à partir de la ligne n°2, chaque ligne du fichier correspond à un produit alimentaire unique
    - pour chacune de ces lignes les informations sont séparées par des |
    - Certains attributs comme ingrédients, additifs et allergènes contiennent plusieurs informations séparées par des virgules.

Instructions

- Créez un nouveau projet **STS/GitHub/Maven** nommé **traitement-fichier**
- Pensez à réutiliser les librairies **commons-lang** et **commons-io** utilisées les jours précédents.
- **Organisez les packages comme vous le souhaitez**
- Afin de traiter le fichier voici le **modèle objet préconisé** :

  L'idée est d'avoir une classe, qu'on va appeler Stock, et qui va posséder la liste de tous les produits. A chaque fois que vous traiterez une ligne de fichier et que vous en extrairez un produit, vous l'ajouterez à votre stock.

  Quelle est la structure d'un produit ? Un produit reste assez simple dans sa structure.
  La classe Produit possède :

  une catégorie
  une marque
  un grade nutritionnel (de A à F)
  un tas de données nutritionnelles (sucre, sel, etc..)
  une liste d'ingrédients
  une liste d'allergènes
  une liste d'additifs.

  Les attributs de type catégorie, marque, ingrédients, allergènes et additifs peuvent être traités, soient comme des String, soient comme des objets. Afin de préparer les TP ultérieurs il est préférable d'opter pour les objets.

  - Créez une classe **Ingredient** avec un attribut libelle
  - Créez une classe **Allergene** avec un attribut libelle
  - Créez une classe **Categorie** avec un attribut libelle
  - Créez une classe **Marque** avec un attribut nom
  - Créez une classe **Produit** avec
    - un attribut de type **Categorie**
    - un attribut de type **Marque**
    - un attribut scoreNutritionnel
    - un ensemble d'autres attributs pour stocker les différents valeurs nutritionnelles (énergie, sel, etc…)
    - une liste d'objets Ingredient
    - une liste d'objets Additif
    - une liste d'objets Allergene
  - Créez une classe **Stock** qui contient l'ensemble des produits.

- Créez une classe exécutable **ApplicationOpenFoodFacts** avec un menu permettant
  - de rechercher les meilleurs produits pour une Marque donnée

- de rechercher les meilleurs produits pour une Catégorie donnée
- de rechercher les meilleurs produits par Marque et par Catégorie
- d'afficher les allergènes les plus courants avec le nb de produits dans lesquels ils apparaissent.
- d'afficher les additifs les plus courants avec le nb de produits dans lesquels ils apparaissent.

- **Consignes :**
  - Mettez en place un découpage en classes de service comme vous l'avez vu dans le TP recensement avec une gestion des exceptions.
  - Essayez d'avoir une approche objet. Chaque concept doit avoir un objet associé.
  - Attention, la liste d'ingrédients de certains produits n'est pas séparée par des virgules, ce qui complexifie le traitement des données.

Facultatif

- Afin d'économiser l'occupation mémoire, essayez de réutiliser les ingrédients, allergènes et additifs déjà connus.
  - Par exemple si le « Lait » est présent dans 1000 produits, il serait judicieux qu'il n'y ait qu'une seule instance de cet ingrédient que tous les produits partagent. Cela signifie qu'il faudra sans doute une structure pour stocker les ingrédients déjà connus (Une HashMap ?).
- Ce qui est valable pour les ingrédients l'ait aussi pour les allergènes et les additifs. On économise la mémoire.

Commitez vos développements sur GitHub