R2.01 BUT1 Info

TP 09 - Semaine 17

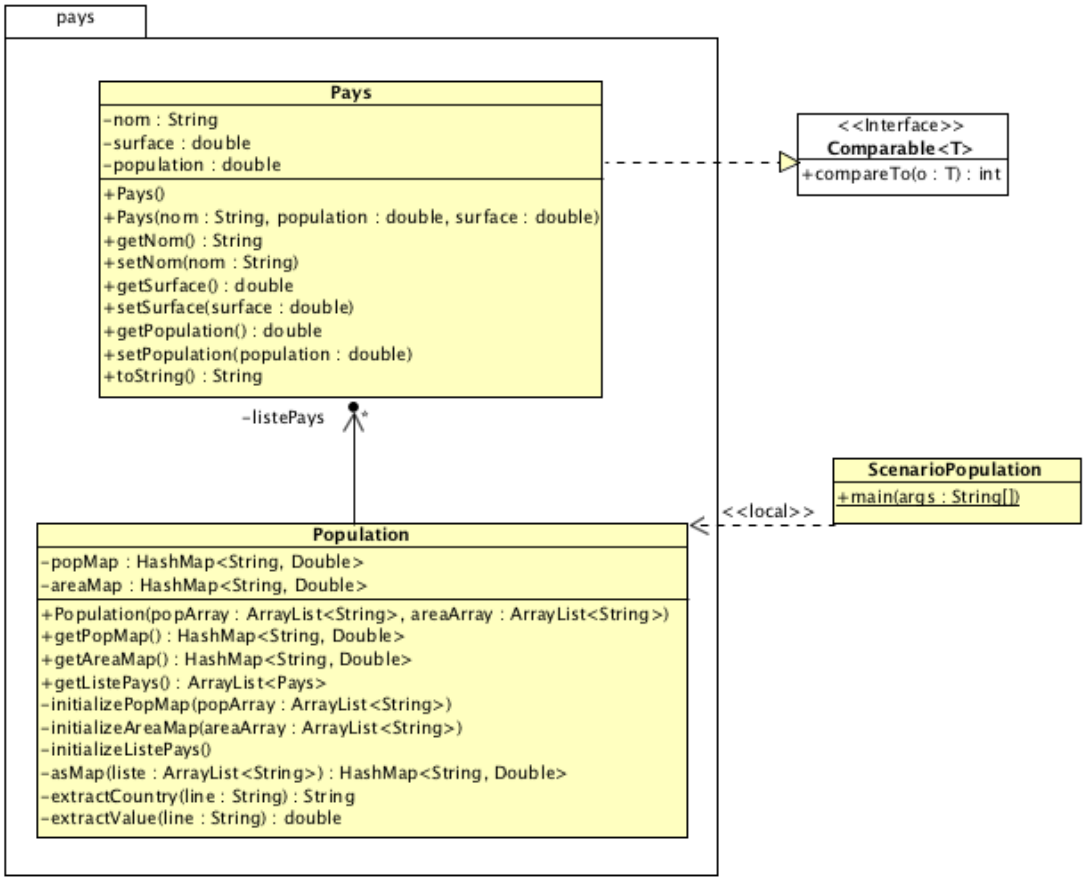
Objectifs du TD-TP

Manipuler des collections (`ArrayList` et `HashMap`)

Préambule : Relecture de cours

Le premier travail est de relire le cours n°8 sur les structures de données de type `HashMap`

La classe Population (1ère partie)



Au TD/TP précédent, nous avons défini et utilisé la classe `Pays` qui est réutilisée ici. Il faut donc reprendre la classe `Pays`.

L'objectif de l'application est de traiter et manipuler des informations sur des pays. L'application correspond au diagramme de classes ci-dessus.

La classe `Population` est une classe qui va réaliser des manipulations sur les données des pays. Elle se trouve dans le package `pays`.

Dans cette première partie, nous vous fournissons des données. La semaine prochaine, vous lirez les données sur les pays directement dans des fichiers.

La classe `Population` permet de créer une liste d'objets de type `Pays`, représentée par un `ArrayList`. Elle possède 3 attributs :

- `listePays` : une collection d'objets de type `Pays` représentée par un `ArrayList<Pays>`

- `popMap` : une collection de couples de données (nom_pays, nb_habitants) représentée par un `HashMap<String, Double>` dont les clés sont les noms des pays et les objets associés les nombres d'habitants.
- `areaMap` : une collection de couple de données (nom_pays, surface) représentée par un `HashMap<String, Double>` dont les clés sont les noms des pays et les objets associés les surfaces en km2.

Le constructeur

Pour ce TP le constructeur de la classe `Population` prend en paramètres deux `ArrayList` : un `ArrayList` (`areaArray`) contenant les surfaces des pays du monde et un autre contenant le nombre d'habitants des pays (`popArray`). Attention ces deux `ArrayList` ne sont pas des attributs : ils seront donnés en paramètre aux méthodes d'initialisation des attributs qui sont eux, de type `HashMap`.

Pour cette semaine, le jeu de données sera un sous-ensemble des données pour 21 pays qui sont définies dans la classe `ScenarioPopulation`**.** Les données dans les ArrayList sont données avec le même format que les données que nous lirons dans les fichiers.

Vous utiliserez la classe `ScenarioPopulation` pour tester la création d'une instance de `Population` et les méthodes que l'on aura écrites. La plupart des méthodes seront privées mais vous pourrez temporairement les mettre en `public` pour les vérifier.

Travail demandé

Nous avons en entrée deux `ArrayList` dont les éléments sont des chaînes de caractères du format suivant :

"Afghanistan 647500"
"Akrotiri 123"
"Albania 28748"
"Algeria 2381740 "

La chaîne contient donc : 1 nom de pays, des espaces et un chiffre.

L'objectif maintenant est de remplir les deux `HashMap` de la classe `Population`.

On ne va pas écrire toutes les méthodes de la classe `Population` cette semaine.

A - Extraction des données dans une chaîne de caractères

1) Pour commencer nous allons écrire un squelette pour la classe `Population` (entête, attributs, constructeur) auquel nous ajouterons au fur et à mesure les méthodes qui permettront l'initialisation des deux `HashMap`. Pour l'instant le constructeur vérifie uniquement la validité des paramètres.

Pour pouvoir remplir les `HashMap` nous avons besoin de pouvoir séparer le nom du pays de la donnée associée.

2) Ecrire une méthode privée:

`private String extractCountry(String line)` : qui prend une chaîne de caractères en paramètre et retourne le nom du pays extrait de la chaîne.

Pour cela :

- Il faut analyser la chaîne caractère par caractère pour trouver l'indice dans la chaîne où commencent les caractères qui sont des nombres.
- Ensuite, on extrait la sous-chaîne qui correspond au nom du pays

– Enfin il faut supprimer les espaces superflus.

Les méthodes nécessaires de la classe `String` :

- `public char charAt(int index)` : Returns the char value at the specified index
- `public String trim()` : Returns a copy of the string, with leading and trailing whitespace omitted.
- `public String substring(int beginIndex)` : Returns a new string that is a substring of this string. The substring begins with the character at the specified index and extends to the end of this string.
- `public String substring(int beginIndex, int endIndex)` : Returns a new string that is a substring of this string. The substring begins at the specified `beginIndex` and extends to the character at index : `endIndex - 1`.

Pour tester si un caractère est un nombre, on va utiliser une méthode de la classe wrapper `Character` :

- `public static boolean isDigit(char ch)` : Determines if the specified character is a digit. Returns: true if the character is a digit; false otherwise.

3) Ecrire une méthode privée :

`private double extractValue(String line)` : qui prend une chaîne de caractères en paramètre et qui retourne le nombre contenu dans la deuxième partie de la chaîne.

Comme pour la méthode précédente, il faudra :

- Repérer l'indice du premier caractère numérique.
- Extraire la chaîne de cet indice jusqu'à la fin de la chaîne.
- Supprimer les espaces pouvant subsister
- Transformer le résultat en double avec la méthode classe `Double.parseDouble(String s)`

`public static double parseDouble(String s)` : Returns a new double initialized to the value represented by the specified String.

4) Mettre temporairement les deux méthodes précédentes avec une visibilité `public`. Vérifier le fonctionnement de vos méthodes avec l'instance « pop» déjà créée dans la classe `ScenarioPopulation` , qui vous est fournie, et en passant en paramètre une chaîne parmi les exemples de surface et population.

B - Utilisation des structures de données type HashMap (voir cours 8)

Le plus simple pour manipuler les données de surface et de population d'un pays est d'utiliser une structure de données permettant d'associer une clé à une valeur, c'est ce que permet de faire la collection `java.util.HashMap` de Java.

Un `HashMap` se déclare avec 2 types :

`HashMap<K,V>` : K le type des clés et V le type des valeurs associées.

Exemple :

```
private HashMap<String, Double> popMap; // Clé de type String et valeur de type Double (attention pas de primitif, seuls les objets peuvent être mis dans des collections)

popMap = new HashMap<String, Double>(); //un objet Double peut être créé à partir d'un double ou d'un objet String ;
```

```
popMap.put("France", new Double(55000000));
popMap.put("Belgium", new Double("45000000"));
```

Transformation d'un ArrayList en HashMap

Maintenant nous devons initialiser chaque `HashMap` à partir des deux `ArrayList` contenant les données :

- popMap doit récupérer les données de popArray
- areaMap doit récupérer les données de areaArray

Pour cela nous avons besoin de convertir un `ArrayList` en un `HashMap.`

- Ecrire la méthode suivante :

`private HashMap<String,Double> asMap(ArrayList<String> liste) :` Cette méthode prend en paramètre un `ArrayList` contenant des objets `String` de la forme "pays donnée " et retourne en résultat un `HashMap` dont les éléments auront comme clé le nom du pays et la donnée en valeur.

Cette méthode utilise les méthodes d'extraction des questions 2) et 3).

Elle crée un `HashMap` et le remplit par une boucle (for each) sur la liste passée en paramètre. Pour chaque élément de la liste, elle extrait le pays (la clé) et la valeur associée.

Elle retourne ce `HashMap` en résultat.

RENDU DU TP

Déposer sur moodle **tout** votre dossier source et la javadoc générée.