

Programmation Orientée Objets avec Java

Chapitre 8-9 : Streams - Prog. Fonctionnelle

Stéphane Malandain / Yassin Rekik
Semestre d'automne 2024

Préparation

Tout d'abord, il est demandé de créer une classe *Individu* ayant comme attributs : une **taille**, une **dateDeNaissance**, un **lieuDeNaissance**, un **poids** et une liste de **hobbys**. Ces attributs sont non modifiables après la construction de l'objet mais sont manipulables via des getters.

Les lieux de naissance possible sont définis dans une énumération adéquate **LieuxDeNaissance** parmi les valeurs TOKYO, PARIS, NAIROBI, LOS_ANGELES, BAMAKO, MOSCOU, CARACAS et TBILISSI. Cette énumération devrait permettre l'affichage agréable du nom de la ville. Les dates de naissance devront utiliser le package **java.time** (\geq Java 8), c'est-à-dire les dates Java 8. Les dates possibles seront entre le 1er janvier 1990 et le 31 décembre 2024 et à n'importe quelle heure de 0h00 à 23h59. Les hobbies sont à prendre dans une énumération adéquate parmi les valeurs JEUX_VIDEO, SPORT, RANDONNÉE, SKI, CINÉMA, CUISINE, LECTURE. Le poids doit être compris entre les valeurs 60kg et 100kg (au kg près). La taille sera comprise entre 150cm et 190cm (au centimètre près).

L'affichage des individus doit être « lisible », comme :

Né le ... à ..., mesure ...cm, pèse ...kg, aime ..., ...,

Il est ensuite demandé de créer une classe **IndividuUtils** contenant deux méthodes statiques :

1. **Individu createIndividu()** permettant d'obtenir un Individu dont les caractéristiques seront tirées au hasard conformément aux valeurs indiquées ci-dessus.
2. **List<Individu> createIndividus(int n)** permettant d'obtenir une liste de n individus « tirés au hasard ».

Exercice 1

Écrire un test (classe **Test1** avec main) permettant, à l'aide de streams à partir d'une même liste de 100 individus tirés au hasard, de :

1. afficher tous les individus nés en 2000 au mois de mars,
2. afficher les 5 premiers individus nés en juillet 1993,
3. afficher le nombre total d'individus nés en 1994,
4. afficher les 5 plus grands individus nés au mois de décembre de l'année 2001,
5. afficher les 5 plus petits individus de taille au moins 180cm nés en 2002,

6. afficher les 5 plus petits individus de taille au moins 180cm nés en 2002 par ordre décroissant de leur taille,
7. afficher la liste (sans redondance) des hobbies des 5 plus lourds parmi les 20 plus grands individus nés en 1999,
8. afficher la liste (sans redondance) des mois de naissance des 20 plus grands individus nés en 1997.

Exercice 2

Dans une classe Test2 avec un main :

- surcharger la méthode **IndividuUtils.createIndividus** de telle sorte que l'on puisse contrôler le type concret de la liste obtenue en résultat, tester avec **ArrayList** et **LinkedList**.
- surcharger la méthode **IndividuUtils.createIndividus** de telle sorte que l'on puisse contrôler la source de création des individus, tester avec un appel à la méthode **IndividuUtils.createIndividu** ou un **Supplier<Individu>** renvoyant toujours un même individu.
- créer une méthode **void test(List<Individu> l)** permettant d'afficher la liste des individus à l'aide d'un for étendu. Pour les questions de l'exercice 1, reprendre celles qui génèrent des listes d'individus et obtenir en résultat une liste, laquelle sera envoyée à la méthode test pour affichage.
- surcharger la méthode **test** de sorte que l'on puisse contrôler l'action de « sortie ». Tester avec une sortie réalisant deux affichages : l'affichage de base suivi d'un affichage customisé de l'individu (par exemple en affichant que ses hobbies).
- créer un **Consumer<Individu>** permettant de stocker des individus dans un fichier. Utiliser ce Consumer de sorte à stocker dans un fichier l'ensemble des individus de la liste initiale (appel à la méthode test surchargée).
- créer un **Supplier<Individu>** permettant de lire les individus stockés dans un fichier. Utiliser ce Supplier pour « injecter » des individus dans la liste créée via **createIndividus** avant les tests.