UNIVERSITÉ D'AVIGNON
ET DES PAYS DE VAUCLUSE

M2 ILSEN – 2019/20

UE Ingénierie du document et de l'information

UCE Indexation & recherche

Vincent Labatut

TP 0 | Présentation et préparation

L'objectif de cette série de TP est d'implémenter des fonctionnalités de base nécessaires à l'indexation. Nous allons donc nous attacher à programmer des versions simples de ces structures et algorithmes, afin de bien comprendre comment ils fonctionnent.

1 Organisation

Outils. Nous utiliserons le langage Java et l'IDE Eclipse. Le fichier Objets Java pour la RI récapitule les différentes classes Java permettant de représenter des collections d'objets. Il est recommandé d'y jeter un il avant de commencer le TP.

Évaluation. L'évaluation des TP est réalisée lors d'un examen correspondant à la toute dernière séance de la série de TP. Cette séance s'organise comme un TP normal, lors duquel on implémente une fonctionnalité vue en cours mais pas déjà traitée lors des TP précédents.

Chaque étudiant doit se présenter au TP noté avec le code source correspondant aux exercices des TP précédents. En effet, le TP noté consiste à se baser sur l'une des versions de l'outil développée lors des séances non-évaluées, et à l'enrichir. Un étudiant qui se présenterait sans cette ressource (ou même avec une version incomplète) n'aurait vraisemblablement pas le temps matériel de finir le TP noté.

2 Corpus

Le TP est fourni avec deux corpus (collections de textes à traiter) : l'un basé sur Wikipedia, et l'autre sur des résumés d'articles scientifiques.

Wikipedia. Le premier est en français et prend deux formes distinctes : `wp` qui est la version complète 2 871 (textes), et `wp_test` qui est un sous-ensemble (200 textes). Dans un premier temps, il est conseillé de déboguer votre code source sur `wp_test`, pour des raisons de rapidité. Lorsque votre programme semble fonctionner correctement, alors vous pouvez l'appliquer à `wp`.

Pour information, le corpus est composé d'articles de la version française de Wikipedia. Ceux-ci ont été obtenus en crawlant ce site à partir de l'article Recherche d'information, de façon interne (i.e. en considérant seulement les liens vers d'autres pages françaises de Wikipedia) et en limitant la distance parcourue à deux hops. Les articles sont nettoyés de manière à ne contenir que les caractères suivants : lettres, chiffres, espace, `'\n'`, et la ponctuation `'():,-!?.";&@%+`.

Springer. Le second corpus, `springer` est en anglais. Il se compose de résumés de 7 823 textes issus de la recherche médicale. Il est lui aussi nettoyé et ne contient que des caractères alphanumériques, ponctuation et espaces. Son intérêt est qu'il est issu d'une campagne d'évaluation, et à ce titre il est fourni avec une référence : un ensemble de 25 requêtes prédéfinies, pour chacune desquelles des experts ont identifié l'ensemble des textes pertinents. Autrement dit, on connaît la vérité terrain pour ces requêtes-là, ce qui permet d'appliquer les mesures et méthodes d'évaluation présentées en cours.

3 Consignes générales

Code source. Les consignes suivantes sont valides pour toute la série de TP. Leur but est de s'assurer une certaine uniformité du code source que vous produisez, afin faciliter votre travail ainsi que votre évaluation lors de la séance de TP notée.

- Il est obligatoire de respecter les contraintes d'implémentation spécifiées dans les sujets : identificateurs (noms de méthodes, paramètres, variables, etc.), types de paramètres et de retour, etc.
- Vous ne devez pas créer d'autres classes et méthodes que celles.
- Chaque méthode secondaire doit être testée dans le `main` de sa classe, tandis que les méthodes principales sont testées à partir d'une classe de test située dans le package-racine (ceci est détaillé dans le sujet).
- Vos test doivent forcément traiter *au moins* les exemples proposés dans les sujets (mais il est préférable de les étendre).
- Pour des raisons de portabilité, il est interdit d'utiliser des chemins *absolus* dans votre code source (seulement des chemins relatifs).
- Chaque classe fournie contient une méthode `main`, que vous devez compléter pour tester les différentes méthodes écrites, et ce *au fur et à mesure de leur production.*

Réponses textuelles. Certains exercices demandent des réponses qui ne prennent pas la forme de code source, mais d'explications à fournir. Vous devez alors créer un fichier `reponses.txt` dans la racine du projet, dans lequel vous placerez les réponses (textuelles) à ces questions.

Durée. Chaque sujet est prévu pour une durée d'1h30. Vous ne parviendrez probablement pas à les terminer dans le temps imparti, ce qui est normal. Ils sont alors à finir chez vous avant la séance suivante. Quand 2 ou 3 séances d'1h30 s'enchaînent, alors vous devez finir les 2 ou 3 sujets de TP concernés chez vous.

4 Préparation

Projets. L'archive disponible sur e-uapv contient deux projets Eclipse. Le projet `Common` contient les deux corpus, placés respectivement dans les dossiers `wp` et `wp_test` pour le premier, et `springer` pour le second.

Le projet `Index1` contient le code source. Il s'agit du squelette de notre outil d'indexation et de recherche. Il contient des classes dans lesquelles les champs sont définis, mais les méthodes sont vides. Il vous faut donc les compléter, en suivant les instructions des sujets. Il est possible que nous développions d'autres versions du logiciel, appelées `Index2`, `Index3`, pour effectuer des comparaisons. Toutes utiliseraient alors elles aussi les données de `Common`.

Importation. Vous devez importer ces deux projets dans Eclipse. Pour cela :

- Dans Eclipse, allez dans *File* > *Import* > *Existing Projects into Workspace*.
- Dans *Select archive file*, sélectionnez l'archive téléchargée depuis e-uapv.
- Dans *Projects*, sélectionnez les deux projets disponibles.
- Vérifiez que l'option *Copy projects into workspace* est bien cochée.
- Cliquez sur *Finish* pour démarrer l'importation. Les deux projets vont alors apparaître dans votre espace de travail.

Chemins. Pour une utilisation normale de l'index, la classe `tools.FileTools` vous fournit des méthodes prédéfinies permettant de récupérer tous les chemins dont vous avez besoin. Ils sont calculés à partir des valeurs contenues dans la classe `Configuration` (du même package). Avant d'utiliser ces méthodes, il est donc nécessaire de spécifier les paramètres du processus d'indexation au moyen des accesseurs fournis par `Configuration`, et en particulier `setCorpusName`, qui vous permet d'indiquer le nom du corpus que vous désirez traiter (`wp`, `wp_test` ou `springer`).

Dans le cadre du test individuel des méthodes que vous développez, il est possible que vous ayez besoin de construire vous-même un chemin, ponctuellement. Notez que sous Eclipse, par défaut, les chemins *relatifs* sont interprétés par rapport au projet en cours d'exécution. Donc, si vous voulez accéder au corpus situé dans le dossier `wp` du projet `Common`, à partir du code source situé dans le projet Index1, vous devez utiliser le chemin relatif suivant : `../Common/wp`.