M1 informatique – Sujet de projet d'architecture logicielle 2015
Bibliothèque de gestion de réglages applicatifs

F. Nicart

7 décembre 2015

1 Modalités

— L'implémentation et le compte-rendu sont à rendre avant le **4 janvier 2016** sans délai supplémentaire.
— Vous constituerez des groupes de 2 **personnes maximum**.
— Vous livrerez votre projet sous forme d'une archive en utilisant FileX si nécessaire à :
`florent.nicart@univ-rouen.fr`

2 Présentation du projet

Une application a souvent besoin de mémoriser certains paramètres d'exécution définis ou modifiés par l'utilisateur de façon persistante. On se propose de produire une bibliothèque permettant de charger, lire, modifier, ajouter et enregistrer des réglages de telle sorte que chaque partie de l'application puisse accéder directement aux réglages qui l'intéresse.

Les réglages seront enregistrés dans un simple fichier.

Un réglage pourra être une valeur de trois types seulement : chaîne de caractères, entier ou réel. Les réglages seront identifiés par une clé dite *clé de réglage* qui sera une simple chaîne de caractères.

Les réglages pourront être structurés en *groupes*. Ces derniers peuvent être imbriqués indéfiniment et sont également identifiés par une clé dite *clé de groupe*. Les clés peuvent être concaténées à l'aide d'un symbole "." pour former des chemins absolus ou relatifs. Par exemple :

```
calcul.algo1.maxIter // Identifie un réglage de type entier "maxIter"
calcul.algo1         // Identifie le groupe de réglages "algo1"
algo1                // Identifie le groupe de réglages "algo1" lorsqu'appliqué depuis le
                     // groupe "calcul"
maxIter              // Identifie un réglage de type entier maxIter lorsqu'appliqué depuis
                     // le groupe de réglages "calcul.algo1"
```

Dans la suite, nous ne ferons pas de distinction entre clés absolues ou clés relatives.

Toute application utilisant la bibliothèque pourra :
— demander le chargement de l'ensemble des réglages à partir d'un fichier,
— demander la sauvegarde de l'ensemble des réglages vers un fichier,
— obtenir une valeur à partir à partir de sa clé de réglage,
— ajouter ou modifier une valeur à partir à partir de sa clé de réglage et d'une nouvelle valeur,
— supprimer une valeur à partir à partir de sa clé de réglage,
— obtenir une référence à un groupe à partir sa clé de groupe,

— supprimer un groupe à partir sa clé de groupe,
— accéder à un groupe ou une valeur de manière relative, c'est à dire à partir d'un autre groupe,
—

On produira trois clients distincts

— `display` : un programme prenant en entrée (paramètre de ligne de commande) le nom d'un fichier de réglages et affichera le contenu de ce dernier de manière indentée : les réglages seront affichés sous la forme `clé = valeur (type)` (où type indique le type de la valeur, tandis que chaque groupe donnera lieu à l'affichage de son nom et une tabulation (supplémentaire) sera ajoutée devant chaque ligne correspondant à son contenu.
— `edit` : un programme prenant en entrée le nom d'un fichier de réglages et permettant de naviguer dans son contenu, de modifier, ajouter ou supprimer des réglages ou des groupes. Ce programme pourra être implémenté soit sous forme graphique avec *swing*, soit en ligne de commandes.
— `demo` : un programme de démonstration validant chaque fonctionnalité.

Contraintes de réalisation :

— Le format de sauvegarde est laissé à votre discrétion (exemple XML, JSON, ou format de votre conception).
— On suppose que la bibliothèque pourra offrir à l'avenir des implémentations internes différentes.

3 Travail à rendre

Vous fournirez une archive contenant :

— toutes les sources de votre projet. Vous vous assurerez que celles-ci se compilent et s'exécutent parfaitement sans erreur (un projet qui ne compile pas ne sera pas évalué). La réalisation doit être structurée en gardant à l'esprit une distribution publique de votre code.
— la *javadoc* de votre bibliothèque que vous aurez relue au préalable avec un esprit critique. On doit pouvoir la considérer comme l'*API*[1] de votre bibliothèque.
— un rapport décrivant précisément, à l'aide de patrons de conception, l'architecture de votre bibliothèque et des applications. Vous fournirez un ou plusieurs diagrammes UML illustrant ceux-ci. Pour chaque patron employé, vous indiquerez le rôle de vos classes qui y participent. Vous consignerez également votre analyse architecturale en motivant l'emploi de chacun des patrons utilisés, mais aussi en expliquant, par exemple, pourquoi vous avez choisi de ne pas appliquer certains patrons potentiellement applicables.
Ce rapport a pour but de mettre en évidence vos compétences d'architectes logiciels et aura autant de poids dans l'évaluation que le programme lui-même.

Dans le cas où votre archive dépasserait la taille de 1 Mo, vous êtes invités à utiliser le service de stockage *FileX* accessible depuis votre *ENT*. (Pensez à régler la date d'expiration à au moins 30 jours)

1. Application Programming interface.