Projet Mini-Éditeur
Master Informatique

Gerson Sunyé
gerson.sunye@univ-nantes.fr

1 Objectifs

Les objectifs de ce projet sont les suivants :

1. présenter une technique simple de conception à objets ;
2. montrer l'utilisation de différents patrons de conception ;
3. montrer la construction parallèle des diagrammes statiques et dynamiques en UML ;
4. montrer un exemple pratique de mise en œuvre en langage à objets de diagrammes UML.

La conception sera réalisée pendant les séances de travaux dirigés et l'implémentation pendant les séances de travaux pratiques. Il constitue le travail pratique noté du module Génie Logiciel.

L'outil de conception conseillé est Papyrus UML, sur Eclipse. Les langages de programmation préconisés sont Scala, Kotlin ou Groovy.

2 Sujet

Il s'agit de construire un mini-éditeur de texte offrant les concepts et les fonctions suivants :

— le texte est contenu dans un *buffer* (zone de travail) ;
— il existe une notion de sélection de texte, avec des actions utilisateur permettant de déplacer le début et la fin de la sélection ;
— copie de la sélection dans un *presse-papier* ;
— copie de la sélection dans le presse-papier puis effacement de la sélection ;
— remplacement (collage) de la sélection par le contenu de presse-papier ;
— possibilité de grouper des actions utilisateurs dans des macros ;
— plusieurs interfaces homme-machine peuvent être associées à un même buffer.

La mise en oeuvre du mini-éditeur doit comporter :

1. les actions d'édition de base et les macros ;
2. l'enregistrement les actions de l'utilisateur et la possibilité de les rejouer ;

3. l'annulation en la répétition d'actions, avec une capacité quelconque dans l'annulation (autrement dit on peut revenir au début).

3 Travail à rendre

Le travail sera fait par des groupes d'au plus 2 personnes. Chaque groupe déposera ses livrables sur le serveur Madoc : `http://madoc.univ-nantes.fr/`. Les livrables envoyés par mail ne seront pas pris en considération. Les rapports seront rendus en format PDF, accompagné des sources (.tex, .doc, .jpg, etc.). Le code source et les tests seront rendus en un fichier archive `tar.gz`, qui contiendra obligatoirement un fichier Maven (pom.xml) de construction/configuration du projet. Les programme exécutables ainsi que les fichiers intermédiaires de compilation (*.class, *.pyc, *.aux, *.o ; *.c, *h, etc.) ne doivent pas être rendus avec le code source.

Le projet sera rendu en 1 seul livrable contenant le dossier de conception et le code source Scala correspondant. Le dossier de conception doit être assez complet pour permettre la mise en œuvre de la solution sans faire appel à des solutions magiques. Il doit comprendre la conception des différentes parties du mini-éditeur : zone de travail, presse-papier, copier/coller, enregistrement d'actions, etc.

4 Conseils

— N'organisez pas votre dossier de conception en fonction d'UML. L'objectif de la conception est de proposer une solution à un problème donnée, ce n'est pas de présenter les différents diagrammes UML.
— Plutôt qu'un seul grand diagramme de classes, dessinez plusieurs petits diagrammes liés et expliquez les choix/compromis de conception.
— Chaque diagramme de conception UML présente un aspect différent du même modèle. Et les aspects doivent être cohérents entre eux.
— Seul le dossier de conception du logiciel a été demandé. Choisissez donc les éléments de modélisation adaptés à la conception.
— Restez simple. Et précis. Le modèle doit être clair, avant tout autre chose.
— Le modèle de conception doit être assez détaillé pour qu'un étudiant de master informatique soit capable de le mettre en œuvre.
— Évitez les banalités. Priorisez la qualité à la quantité.

5 Échéance

— Le 8 décembre 2014 à 23h30.