R6.A06.D05 : Maintenance applicative
Feuille TD-TP n° 1

Refactoring sur IntelliJ – Prise en main

Objectifs :

1.- *Reprise en main de l'outil de développement utilisé : IntelliJ + Java*
2.- *Découverte des outils de l'IDE destinés au refactoring (et inspection de code préalable)*

Ressources à votre disposition :

- L'archive **`junit5-jupiter-starter-gradle.zip`**

  C'est un projet 'Modèle' minimal pour le développement en Java avec IntelliJ et gradle. La fonction main() de son unique clase Main affiche « Hello world ».

  Il devra être configuré pour le développement demandé.

- Le fichier **`tuto_refactoring_IntelliJ.pdf`**
- L'archive carApp.zip contenant l'application à analyser avec IntelliJ et le plugin CheckStyle.

Préparation du travail

1.- Création du dossier consacré aux TDs et TPs de cette ressource

Dans le dossier de votre espace réseau destiné à vos travaux pratiques, créer un sous-dossier destiné à recevoir tous les TPs de la présente ressource. Nous l'appellerons, par exemple : **`r6_AD`**

Dans ce dossier, créer un dossier **`tdtp1`**.

2.- Installation des fichiers de 2 projets

a. Télécharger l'archive **`junit5-jupiter-starter-gradle.zip`** disponible sur eLearn. Décompresser l'archive.
b. Déposer le dossier décompressé dans **`r6_AD\tdtp1`** puis supprimer l'archive **`.zip`** téléchargée.
c. Renommer le projet, par exemple **`tuto-refactoring`**
d. Lancer IntelliJ, ouvrir le projet **`tuto-refactoring`**
e. Compiler, exécuter **`main()`**

f. Télécharger l'archive **`carApp.zip`** disponible sur eLearn. Décompresser l'archive.
g. Déposer le dossier décompressé dans **`r6_AD\tdtp1`** puis supprimer l'archive **`.zip`** téléchargée.

Vous êtes prêt à travailler.

Vous avez 2 exercices à traiter. Pour chacun d'eux, créer un dépôt afin de tracer/sauvegarder chaque étape dans une nouvelle version de l'application.

Exercice 1 – Tuto refactoring-intelliJ

Travail à faire

Créer un dépôt git pour sauvegarder l'avancement de votre travail.

Objectif de l'exercice :
Dérouler le tuto refactoring-IntelliJ pour vous familiariser avec les techniques de refactoring vues en cours et leur mise en œuvre avec IntelliJ.

Cela suppose :
- Au préalable, créer un Créer un dépôt git pour sauvegarder l'avancement de votre travail.
- Écrire le code indiqué sur le tuto afin de mettre en œuvre le refactoring
- Sauvegarder (commit) la refactorisation avant de passer à la technique suivante. Le message du commit documentera la technique de refactoring mise en œuvre
- A la fin, déposer le lien vers votre dépôt GitHub (public) sur eLearn, sur l'espace prévu à cet effet

Exercice 2 – Analyse application carApp

Travail à faire

Objectif de l'exercice :
Analyser l'application fournie et appliquer les modifications mises en évidence par les outils d'inspection de code utilisés (ceux de IntelliJ + plugin Checkstyle)

Cela suppose :
- Au préalable, créer un Créer un dépôt git pour sauvegarder l'avancement de votre travail.
- S'approprier le code de l'application
- Installer le plugin CheckList
- Paramétrer les règles que CheckStyle devra vérifier : Google ou Sun, via File/Setings/Tools/CheckStyle
- Paramétrer l'inspecteur de code File/Settings/Edit/Inspections pour que l'inspection se focalise sur le CheckStyle et Java
- Inspecter l'application et catégoriser les types d'erreurs identifiées (par les outils ou par vous-mêmes)
- Écrire le code indiqué sur le tuto afin de mettre en œuvre le refactoring
- Passer - les tests associés à ce refactoring
- Sauvegarder (commit) la refactorisation avant de passer à la technique suivante. Le message du commit documentera la technique de refactoring mise en œuvre
- A la fin, déposer le lien vers votre dépôt GitHub (public) sur eLearn, sur l'espace prévu à cet effet