SAE 201 Développement d'une application

Etape 2

Rédigé par Clément LEFÈVRE, Thomas GORISSE,
Antoine FADDA-RODRIGUEZ

Semestre 2

Rapport de lisibilité et de propreté du code

Enseignant : Léo DONATI
Département informatique
Année Universitaire : 2021-2022

Afin d'avoir un code des plus propres, nous avons décidé de suivre et de mettre en place plusieurs règles et conventions.
Ainsi nous avons appliqué d'abord les 3 acronymes :

KISS : Keep it simple and stupid

Afin que notre code soit le plus lisible et le plus simple à comprendre par tous les autres utilisateurs, nous avons rendu nos méthodes les plus simples, effectuant une seule action et les plus claires possible.

DRY : Do not repeat yourself

Afin que notre code soit le plus lisible possible, nous avons attribué à chaque action récurrente une méthode ou une boucle en fonction de nos besoins.

BSR : Boy scout rule

Afin de travailler dans un environnement de travail sain et agréable, chaque personne du groupe a respecté les conventions mises en place. En effet, après chaque commit, tous les membres du groupe décrivent leurs améliorations et respectent leurs branch. L'environnement de travail était donc toujours propre et dès lors que l'on arrivait pour travailler, il suffisait de lire les rapports des derniers commits.
De plus, dès lors qu'un fichier était fini, ou qu'une tâche importante était terminée, le membre du groupe concerné se rendait sur les issues du projet et cochait la tâche qu'il venait de terminer.

De plus, nous avons respecté quelques règles que nous avons mises en place entre nous :

Nommer des variables avec un nom cohérent et compréhensible. Ceci permet avant tout de pouvoir comprendre plus facilement comment la méthode / classe fonctionne et à quoi sert l'attribut.

Nommer des méthodes avec un nom cohérent et compréhensible. Ceci permet également la compréhension des méthodes. L'objectif étant de donner un nom par lequel un utilisateur lambda l'aurait appelé. Exemple *getNom* et pas *returnNomPokemon*. Ainsi, les méthodes possèdent un nom cohérent et logique permettant à n'importe qui de pouvoir l'appeler facilement.

Respect de l'indentation. Afin de rendre notre code lisible par tout être humain, il est important de respecter les règles d'indentations.

Langue des méthodes et variables. Nous travaillons tous entre français, néanmoins, la langue utilisée pour coder est généralement l'anglais. Ainsi, nous avons décidé de nommer nos variables et nos méthodes toutes en anglais. De plus, nous avons dû veiller à ne pas en nommer certaines en français et d'autres en anglais. Exemple *getNom* et pas *ObtenirNom.* Or, les interfaces sont toutes nommées en français.

Commentaires. De sorte à pouvoir comprendre le programme d'un utilisateur à un autre, nous avons décidé de réaliser la JavaDoc de chaque classe. Celle-ci explique clairement le but de la méthode, ce qu'elle retourne ainsi que ses paramètres.

Ordre des méthodes. Afin que notre classe soit lisible, toutes les classes sont ordonnées de la même manière : d'abord les attributs, ensuite les constructeurs,

puis les méthodes implémentant les interfaces (d’abord les getters et les setters puis les méthodes fonctionnelles) et enfin les méthodes nous semblant pertinentes(d’abord les getters et les setters puis les méthodes fonctionnelles).
Gestion des conflits. Afin de minimiser nos chances de conflits, nous avons essayé de ne jamais coder dans le même fichier et/ou à la même ligne. Ainsi, dès lors que quelqu’un travaillait sur une classe, il l'annonçait en renseignant sur quelle méthode il travaillait.