Rapport de TP : Analyse et Test de la Classe RocketPokemonFactory

Farouk BRAHIMI

Objectif du TP

L'objectif de ce TP est d'intégrer une version fournie de l'interface IPokemonFactory à notre code, de la tester à l'aide des tests existants, et d'analyser sa qualité, ses performances, ainsi que sa maintenabilité.

Implémentation de RocketPokemonFactory

- **Initialisation de la carte `index2name`** : Une carte associant les indices des Pokémons à leurs noms, avec des valeurs par défaut comme "Ash's Pikachu" pour les indices négatifs.
- **Méthode `generateRandomStat`** : Gènère des statistiques aléatoires pour les attributs du Pokémon.
- **Méthode `createPokemon`** : Crée une instance de Pokémon en fonction des paramètres donnés. Elle gère les indices négatifs en attribuant des valeurs par défaut élevées et une IV de 0.

Tests de RocketPokemonFactory

Les tests ont été réalisés à l'aide de JUnit pour vérifier le comportement de la classe :

- **Vérification des créations correctes de Pokémon** : Les tests ont validé les noms, indices et valeurs des statistiques générées pour des Pokémons comme Bulbasaur et Vaporeon.
- **Tests avec des valeurs invalides** : Deux erreurs principales ont été identifiées :
    1. Les créations de Pokémon avec des paramètres invalides retournent des instances au lieu de `null`.
    2. Aucune exception n'est lancée pour des valeurs invalides.

Les tests de l'interface ne couvre pas tous les methodes de la classe RocketPokemonFactory ce qui cause du dead code (unreahcable code). Il fallait crée un test pour cette classe spécifiquement.

Le code contenait aussi des erreurs detecter par checkstyle.