Rapport de Revue de Code

1. Intégration et Tests

L'implémentation de RocketPokemonFactory a été intégrée dans le projet et testée avec la suite de tests existante. Voici les résultats :

Tests Passés : Tous les tests existants ont été exécutés avec succès.

Tests Échoués : Aucun test n'a échoué.

2. Revue de Code

Points Positifs : Utilisation de UnmodifiableMap pour rendre la map index2name immuable.

Implémentation de la méthode createPokemon conforme à l'interface IPokemonFactory.

Défauts Identifiés :

Performance :

La méthode generateRandomStat utilise une boucle de 1 000 000 itérations pour générer une statistique aléatoire. Cela est extrêmement inefficace et peut ralentir l'application. private static int generateRandomStat() {

int total = 0;

for(int i = 0; i < 1000000; i++) {

Random rn = new Random();

int r = rn.nextInt(2);

total = total + r;

}

return total / 10000;

}

Suggestion : Utiliser une méthode plus simple et efficace pour générer des statistiques aléatoires.

private static int generateRandomStat() {

Random rn = new Random();

return rn.nextInt(100);

}

Utilisation de Random : La création d'une nouvelle instance de Random à chaque itération est inefficace. Il est préférable d'utiliser une seule instance de Random.

private static final Random RANDOM = new Random();

private static int generateRandomStat() {

return RANDOM.nextInt(100);

}

Gestion des Index :

La méthode createPokemon utilise des valeurs par défaut pour les Pokémon avec des index négatifs, ce qui peut ne pas être souhaitable. if(index < 0) {

attack = 1000;

defense = 1000;

stamina = 1000;

iv = 0;

} else {

attack = RocketPokemonFactory.generateRandomStat();

defense = RocketPokemonFactory.generateRandomStat();

stamina = RocketPokemonFactory.generateRandomStat();

iv = 1;

}

Suggestion : Ajouter une validation pour les index négatifs et lever une exception si l'index est invalide. Couverture des Tests :

Les tests existants ne couvrent peut-être pas tous les cas d'utilisation, notamment les index négatifs et les valeurs par défaut.

Conclusion

L'implémentation de la Team Rocket fonctionne correctement avec les tests existants, mais présente des défauts de performance et de gestion des index. Il est recommandé d'améliorer la méthode generateRandomStat et d'ajouter des validations pour les index négatifs. De plus, il est conseillé d'ajouter des tests supplémentaires pour couvrir ces cas.