Rapport de revue de code et tests pour la classe RocketPokemonFactory

1. Introduction

Dans ce rapport, nous présentons nos conclusions suite à l'intégration de la classe RocketPokemonFactory dans notre projet. Nous avons effectué une revue de code et exécuté une suite de tests unitaires pour évaluer la qualité de cette implémentation.

2. Revue de code

La classe RocketPokemonFactory implémente l'interface IPokemonFactory. Elle définit une méthode `createPokemon` pour créer des instances de la classe Pokemon. La classe utilise également une méthode statique `generateRandomStat` pour générer des statistiques aléatoires.

Voici quelques observations concernant le code de la classe RocketPokemonFactory :

- La classe utilise une Map statique et immuable `index2name` pour mapper les index aux noms de Pokémon. Cette approche garantit que les données de mappage ne peuvent pas être modifiées après la création initiale de la Map.
- La méthode `generateRandomStat` génère des statistiques aléatoires comprises entre 0 et 100 en ajoutant 1 000 000 de nombres aléatoires (0 ou 1) et en divisant le total par 10 000. Bien que cette approche fonctionne, il serait plus efficace de simplement générer un nombre aléatoire entre 0 et 100 directement.
- La méthode `createPokemon` gère différents cas en fonction de l'index du Pokémon et génère des statistiques différentes pour les Pokémon spéciaux (comme Ash's Pikachu).

3. Tests unitaires

Nous avons créé une suite de tests unitaires pour la classe RocketPokemonFactory. Les tests vérifient les points suivants :

- Le bon fonctionnement de la méthode `createPokemon` pour les différents types de Pokémon (Bulbasaur, Ash's Pikachu, MISSINGNO).
- La vérification que les statistiques générées par la méthode `generateRandomStat` sont comprises entre 0 et 100.
- La vérification de la création de pokémons avec index négatif.

4. Conclusions

La classe RocketPokemonFactory est correctement implémentée et fonctionne conformément aux spécifications. Cependant, il y a quelques points d'amélioration possibles :
- la méthode createPokemon ne permet pas de gérer les pokemons avec un index négatif.
- On observe que tous les pokémons sauf Ashs pikachu ont des statistiques aléatoires.
- Optimiser la méthode `generateRandomStat` pour générer directement un nombre aléatoire entre 0 et 100.

- Ajouter davantage de Pokémon à la Map `index2name` pour étendre la prise en charge de l'ensemble du Pokédex.

En somme, la classe RocketPokemonFactory a passé avec succès notre revue de code et notre suite de tests unitaires. Nous recommandons d'intégrer cette implémentation dans le projet après avoir apporté les améliorations suggérées.