Spring Exercise

Pokkemon IV/Dust Calculator / Management System

Patrick Geudens

2019-10-07

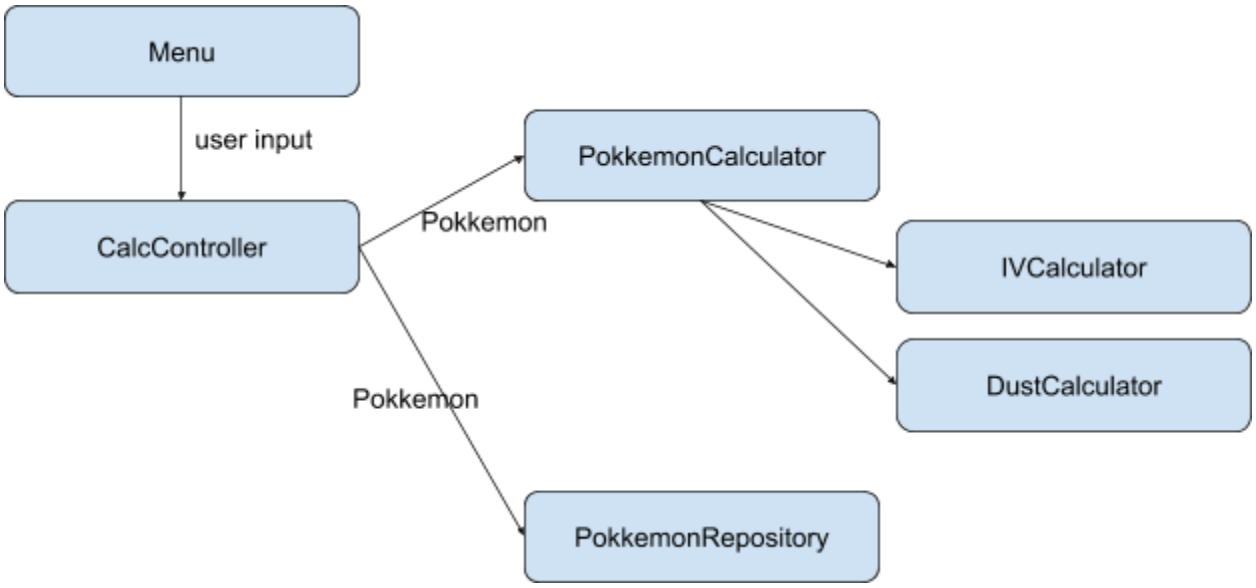
Introduction

Pokemons vangen is een zware bezigheid.. Veel tijd is besteed aan het plannen van uitstappen, polijsten van pokeballen, groentjes snijden voor poke-voer, enzovoort. Dus veel tijd om de IV (Individual Values) en dust requirements te berekenen is er niet.

Dit is waar jullie bedrijf zijn slag gaat slaan, een IV/CP calculator...

Onze Project Architect heeft al een initiele analyse gemaakt van wat we willen maken. Maar heeft zelf geen zin om de code te schrijven. Die taak is voor jullie.

Overview



The Characters

We beginnen met een simpele applicatie en zullen gaandeweg de applicatie uitbreiden met meer functionaliteit en vooral flexibiliteit. Wat je zeker niet uit het oog mag verliezen is dat voor de IVCalculator we de echte formule niet kennen, maar voorlopig een 'placeholder' calculator zullen schrijven die later zal vervangen worden door een calculator met de echte formule. (Je mag nu al weten dat de 'echte' calculator meerdere implementaties zal hebben, dus maak de nodige voorzieningen om dit mogelijk te maken.)

De initiele versie van de applicatie zullen we besturen via de **console**. We gaan voor volgende versies een web frontend schrijven.

Class Menu:

De class Menu is verantwoordelijk voor de interactie met de gebruiker. Hij zal een menu tonen en is daarnaast ook verantwoordelijk voor het vragen van input.

Voor het menu zijn opties toont, vragen we aan de gebruiker zijn:

- Naam
- CharacterLevel

Het initiele menu voorziet de volgende mogelijkheden:

1. Lijst van pokemons
2. Ingeven pokemon
3. Afsluiten applicatie

Output:

- Optie 1: toont een lijst van alle pokemons in de database.
- Optie 2: Vraagt alle informatie over onze pokemons op, namelijk het level van de pokemon, level van de trainer en het 'ras' van onze pokemon (bijvoorbeeld, Pikachu, Snorlax, Bulbarsaur, ... (kijk zelf even op het internet welke mogelijkheden er zijn))
  Na het ingeven van de informatie berekent onze applicatie de IV en dust requirements (meer hierover hieronder) en toont de berekende waarden.
  Indien de gebruiker een PokemonType (ras) ingeeft dat nog niet in de database bestaat worden daar ook de gegevens voor gevraagd.
  De beschrijving van de gegevens kan je vinden in de beschrijving van de PokkemonRepository.

- Optie 3: Sluit de applicatie mooi af.
  Sluit de OutputStream/Connection/EntityManagerFactory mooi af en ruimt alles op.

CalcController:

De CalcController class start de ApplicationContext op en is verantwoordelijk om de input vanuit het menu door te geven naar de services (Calculator en Repository).

PokkemonCalculator

De PokkemonCalculator neemt Pokkemon objecten aan van de CalcController en geeft deze aan de IVCalculator en de DustCalculator om de nodige berekeningen te maken.

IVCalculator

De IVCalculator neemt een Pokkemon object en maakt de nodige berekeningen. Dit is een fictieve berekening en zal later vervangen worden door het correcte algoritme. Wat we al wel weten is dat voor de correcte implementatie er meerdere versies van het algoritme zijn die afhankelijk zijn van de player level en nog 1 andere factor die we nog niet kennen. Hou rekening met deze informatie bij het 'configureren' van je applicatie.

De voorlopige invulling van het algoritme is:

- Neem de base values van het PokkemonType en maak de som
- Trek deze som af van de CombatPower (CP) van de pokemon
- Deel dit resultaat door 3
- Verdeel de overgebleven rest van de deling over Attack, Defense en Hitpoints (in die volgorde)

DustCalculator

Als een speler het level van zijn Pokkemon wil verhogen heeft hij Dust nodig. De hoeveelheid is afhankelijk van het huidige level van de pokkemon. De maximum level van de pokemon is het level van de speler + 1.

De taak van de DustCalculator is om uit te rekenen hoeveel dust er nodig is om de Pokkemon naar zijn maximum level (speler level + 1) te brengen.

We gebruiken de volgende regels:

- Level 1 tot 5: 100 dust per level
- Level 5 tot 10: 250 dust per level
- Level 10 tot 20: 1000 dust per level

- Level 20+ : per 10 levels gaat de dust per level omhoog met 500 dust. Dus vanaf level 20 hebben we 1500 dust nodig, vanaf level 30 2000 dust, vanaf 40 2500.
- Om van level 99 naar level 100 te gaan hebben we 10 000 dust nodig

PokkemonRepository

De repository is verantwoordelijk om de gegevens op te slaan. Er zijn 2 classes die opgeslagen worden:

- Pokemon
- PokemonType

PokemonType: zijn de algemene gegevens per PokemonType en bevat de volgende gegevens:

- Typename (bv. Diglet, Charizard, Mew,...)
- Beschrijving
- BaseAttack (numerieke waarde voor aanval)
- BaseDefense (numerieke waarde voor verdediging)
- BaseHitpoints (numerieke waarde voor de hitpoints)
  Alle numerieke waarden zijn gehele getallen (dus zonder comma)

Pokemon: de gegevens per individuele Pokemon

- PokemonName (naam van de pokemon)
- PokemonType ('ras' van de pokemon)
- Level (numerieke waarde)
- IVAttack (deze waarde berekent onze applicatie dus die vragen we niet aan de gebruiker)
- IVDefense (ook berekent, dus niet aan de gebruiker vragen)
- IVHitpoints (ook berekent, dus niet aan de gebruiker vragen)
- CombatPower (algemene waarde van combat-readiness van onze pokemon)

Voorlopig mag je zelf kiezen hoe je deze gegevens opslaat in de repository. Maar zorg ook hier dat je je gekozen implementatie makkelijk kan vervangen.

Acceptabele oplossingen voor opslag zijn:

- Serialisatie
- JDBC + DB
- JPA + DB