**Plan tworzenia**

1. stworzyć mechanizm do zarządzania atrybutami jednostek
2. stworzyć bazę danych z wszystkimi jednostkami
3. stworzyć pole bitwy

**Wytyczne:**

1. Nazewnictwo klas: AbstractX,XImpl, X (interfejs)
2. Nazwy pakietów:
   1. units
      1. api,
      2. impl,
         1. models,
         2. base

**TODO**

XXX – nie zrobione, XXX – do opisania, XXX – done!

1. SPRINT: Baza danych i jej obsługa
   1. Stworzenie bazy danych z informacjami o jednostkach
   2. Stworzenie modeli

// Setup - tabele z domyślnymi wartościami

UNITS\_SETUP

Id, id\_description, id\_progress, id\_resistances, id\_attack, id\_movment

UNITS\_SETUP\_DESCRIPTION

name, note, race, type

UNITS\_SETUP\_PROGRESS

currentXp, nextLvlXp, level

UNITS\_SETUP\_RESISTANCES

hitPoints, meele, meele, projectile, elemental, mind

UNITS\_SETUP\_ATTACK

source, accuracy, damage, initiative, range, targets

UNITS\_SETUP\_MOVEMENT

stamina, specialPoints

1. SPRINT: Atrybuty
   1. Refaktor – AbstractAttribute/public void update( - wywalic
   2. **Zapisywanie jednostek w bazie danych**
   3. Zapisywanie wartości atrybutów liczbowych jako zmiennoprzecinkowe
      1. zmniejszy to liczbę błędów w przyszłości,
   4. dodanie walidatorów
   5. odporności
      1. podpięcie generatorów zmian wartości atrybutów powiązanych do:
         1. procesu wczytywania danych
         2. Klas atrybutów prostych
      2. usunięcie wytrzymałosci pancerza > zmiany w spec, grafach,
   6. wartości
      1. sprawdzanie wartości przed ustawieniem
      2. dodanie zmniejszania poniżej 0% statystyk
      3. każdej jednostce zapisanej w bazie dać 200%HP
      4. stałe współczynniki przyrostu HP: wojownicy +50, magowie+30, strzelcy +40,
   7. Refaktor - rodzaje atrybutów:
      1. Opisowy,
      2. Liczbowy,
      3. Procentowy,
   8. wczytywanie domyślnych wartości współczynników zmiany
      1. **loadAdditionalData > zrobic kopiowanie atrybutu hitPoints dla maxHp**
      2. **dodanie pliku json opisującego wszystkie zależności między atrybutami**
2. SPRINT: Levelowanie jednostek
   1. Przy nowym poziomie powinny się zmniać wspołczynniki zmiany
3. SPRINT: Buffy / um. Pasywne
   1. Stworzenie języka do zapisu buffów
   2. Stworzenie dynamicznego mechanizmu do buffów
   3. usunąć podział na pozytyne negatyw’’ne
   4. stworzyć uniwersalny obiekt Action, który będzie opisywał oddziałowanie między jednotkami na polu bitwy,
   5. dodanie defaultowych buffów
      1. odporności wrodzone (np. dla nieumarłych buffem będzie odpornosc na zywioly, smierc – zawsze 75%) + sprawdzanie
4. SPRINT: Akcje – Atak
   1. Przebieg
      1. trafi / nie trafi
      2. sprawdź buffy chroniące
      3. zadaj obrażenia
      4. sprawdź życie jednostki
      5. nałóż buffy - kary
   2. Klasa akcji: Action
      1. Typ: attack,
      2. Mapa: nazwa trybutu – zmiana
5. SPRINT: Akcje - Ruch
   1. Przebieg
      1. sprawdź buffy „dobre”
      2. ... „złe”
      3. zaznacz jednostkę,
      4. wykonaj ruch,

* **Przemyśleć**
  + Po co dzielić jednostki na typy? Co daje taki podzial?
  + Celność - co z nią zrobić?
    - Zamienić na szansę na blok/unik/zasłonę,
  + Ciosy: szybki (mniejsze szanse na zasłonę/użycie um. pasywnych) i ciężki (szansa na zadanie obrażeń krytycznych)
  + 3 tryby akcji na polu bitwy:
    - Pass
    - Ofensywny – szansa na zadanie obrażeń krytycznych,
    - Defensywny – większy % na uniknięcie obrażeń + mały % do pancerza
* **Do zrobienia**
  + Model *Akcji:*
    - AttributeChange
    - [] Buff
  + Pętla - Mechanizm atakowania
    - Zapisz id obecnie wybranej jednostki
    - Wybierz jednostkę do zaatakowania
    - Zapisz jej id
    - Wyślij atakującemu #ref do jednostki
    - Wyczyść stan gry