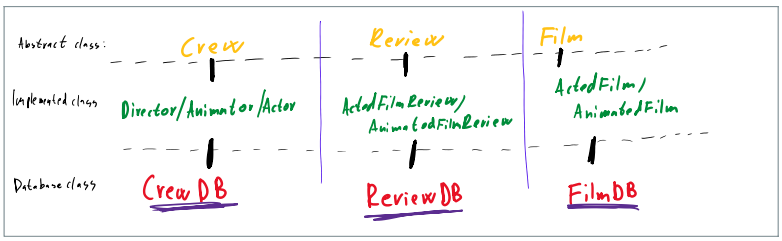
Classes and types

pátek 7. dubna 2023 9:46

Data classes

Abstract class:	Crew	Review	Film
Implemented class	Director/Animator/Actor	ActedFilmReview/ AnimatedFilmReview	ActedFilm/ AnimatedFilm
Database class	CrewDB	ReviewDB	FilmDB

* VŠECHNY DB CLASSY IMPLEMENTUJÍ ROZHRANÍ SQLDataProvider



Crew/Review/Film DBs

- obsahuje Dynamickou strukturu svých typů (Hashtrap) ani?
- funkce pro práci se záznamy, příprava dat z/do SQL

SQL HANDLER CLASS

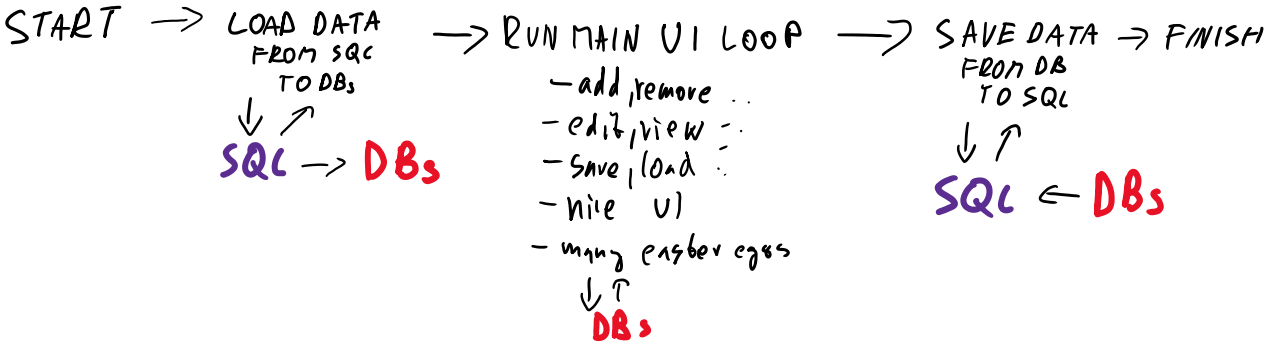
- zprostředkovává načítání/ukládání dat z/do SQL z/do interní databáze programu
- využívá metod DB class (Review, Crew, Film) z interface SQLDataProvider

SQL DataProvider - interface

- metoda pro vrácení všech dat z DB ve formě seznamu (setu?) -> pro SQL
- metoda pro uložení všech dat do DB ze seznamu (setu?) -> z SQL

Structure

pátek 7. dubna 2023 10:16



SQL tables

pátek 7. dubna 2023 10:30

Review

id	INT
film_id	INT
score	INT
comment	TEXT

Film

id	INT
name	TEXT
year_out	SMALL_INT ?
recommended_age	TINY_INT

Crew

id	INT
name	TEXT
[film_id or	INT] ??
(film_ids	TEXT/JSON)
Occupation	TINY_INT ?

– v tabulce crew
budou mít herci svíce rolemi
hned několik záznamů

Jak řešit strukturu interní databází programu

pátek 7. dubna 2023 17:19

1. možnost

Jednoduché řešení pouze 1 DB → FilmDB
- ale některé funkce složitější

Film
- name
- recom.age
- release year
- director
- list_of_actors
- list_of_reviews - Reviews

HashMap of CrewMembers

Více následuje OOP Javy

PROs/CONs
- jednoduchá vnoření struktura — OK
  - MÁME POUZE FilmDB
    - v ní jednotlivé Filmy
      - v každém Filmu seznam herců a hodnocení
- vytváření, práce s filmy — práce s jednou DB — OK
- ukládání do SQL — nutno každý film samostatně rozebrat a poukládat — OK, mírně horší jak 2.
- Načítání z SQL — select nad jedním filmem v review/crew table → převod do interních DB — OK, mírně horší jak 2.
- výpis herců/animátorů svíce filmy → nutno projít všechny filmy
  - mít temporary strukturu pro ukládání zjištěných informací
  - nebo mít navíc databáři všechhorů a postupně nad ní používat dolní funkci, postupně vypisovat — zbytečně složité, ale jde to
- výpis filmů s konkrétním hercem → nutno projít všechny filmy a zjistit jestli hrál taky nebo ne — OK

2. možnost

složitější, ale možná výhodnější řešení — funkce lehce implementovatelné (takhle bych to udělal v C)

MÁME 3 Databáze se vzájemnými odkazy

Něco jako „Simulace" SQL uvnitř Javy → ALE MNOHÉM VĚTŠÍ VYUŽITÍ PAMĚTI ASI

FilmyDB
- list or HashMap of Films
- name
- recAge
- release year
- linksToReviews
  - jen id otsla
- linksToCrew
  - jen id otsla

ReviewDB
- list or HashMap of Reviews
  - id
  - film_id
  - score
  - comment

CrewDB
- list or HashMap of CrewMembers
  - id
  - film_id
  - name
  - occupation

možný přídavek, asi ne
- zjednodušení práce

PROs/CONs
- větší pamětové nároky cca, možná zanedbatelné
- možná výhodnější pro rozsáhlejší struktury a databáze
  - jednodušší implementace některých funkcí
- vytváření a práce s filmy — OK
- ukládání do SQL — OK – interní DB kopíruje, easy SQL TABULKY
- načítání z SQL — OK – ||– , easy
- výpis herců/animátorů svíce filmy
  - získat z CrewDB — OK – lehké rozhodně ale ne the best (určitě lepší než v 1.)
  - jména pak nad nimi procházet
- výpis filmů s konkrétním hercem — OK –