Poznámky k implementaci

Aplikace je naimplementovaná v rámci možností, které daný časový interval umožňoval. Lze tedy mnoho věcí vylepšit. Zaměřil jsem se na splnění požadavků, na funkčnost a na základní úroveň zdrojových souborů. Jistě je prostor pro refaktoring, ale na to nezbyl čas. Momentální stav je snad alespoň postačující.

Použil jsem pouze Javu 7 bez dalších knihoven třetích stran. Nepoužívá se tedy ani maven, ani ant. Programoval jsem v IntelliJ Idea 13.

Testoval jse pouze na Windows 8.

Pro základní použití slouží návod dodaný společně s aplikací.

Pro pravidelné výpisy a čtení vstupu jsem použil ScheduledExecutorService. Nepoužil jsem vlastní časování vláken, ani Quartz a další. V praxi jsem to ještě nepoužíval, tak jsem si to alespoň vyzkoušel.

Následují poznámky k implementaci. Pokud píši, že něco chybí nebo není bez uvedení důvodu, tak to bylo kvůli času.

Poznámky:

- Chybí testy!
- Nepočítá se s tím, že by výpis měl trvat déle, než nastane čas následujícho výpisu. Stejně tak u čtení vstupu. To by mělo být ošetřeno voláním tasku příkazem scheduleAtFixedRate(), který zajišťuje, že v jednu chvíli nepoběží task ve více vláknech. Z toho důvodu je některé použití synchronized nadbytečné, ale není to nic proti ničemu.
- V zadání se jasně píše, že externí soubor lze načíst pouze při startu. Takže později to není možné, i když by se to mohlo hodit. Buď načíst soubor příkazem nebo pravidelně číst soubor(y) v nějakém umístění.
- Počítá se s tím, že se bude pracovat s příkazovým řádkem. Takže pokud není dostupná konzole, tak se program ukončí.
- Ukázkový výpis není podle abecedy, ale přišlo mi vhodné ho seřadit, takže jsem to tak udělal. Snad je to tak v pořádku.
- Nezabýval jsem se problémy spojenými s kódováním. Jak při vstupu a výstupu z konzole, tak i např. při parsování BigDecimalu. Desetinná čárka je tečka.
- Neřešil jsem, když uživatel zadává vstup a během této doby nastane pravidelný výpis. Výstup se jednoduše provede uvnitř vstupu. Vstup lze však dokončit a pokud je validní, tak se uplatní.
- Ve vstupním souboru se vynechávají prázdné řádky. Pokud je na nějakém řádku něco jiného, než co odpovídá validnímu formátu platby, tak se načítání souboru ukončí. Předpokládám, že nekonzistentní stav nikoho nezajímá.
- Požadavky na měnu v souboru jsou, aby byla velkými písmeny. Takže je to tak i v konzoli.
- Nejprve jsem uchovával pouze aktuální stavy kont pro každou měnu (v Map). Tím se nemusí platby stále přepočítávat, ale po provedení platby se platba ztratila. V zadání se píše, že program má uchovávat záznamy plateb, takže jsem přidal, ještě zvlášť kolekci (List), která uchovává platby, tak jak byly zadávány. Existuje kontrola, aby záznamy v obou kolekcích byly

konzistentní. Pokud by mělo dojít k nekonzistenci, tak se program ukončí. Možnost vypsat si platby není implementovaná.
- Není jasně odděleno, aby jádro programu fungovalo nezávisle na externím IO. To by samozřejmě bylo vhodné, kdyby se komunikace s konzolí změnila na něco jiného. Pracovalo by se tedy klasicky s definovanými interfaces. Do konzole by se mohlo vypisovat pouze z jednoho místa atd.

Vynecháno:

- Logování do externích souborů.
- Externí konfigurace.
- Další formáty vstupu a výstupu plateb.
- Výstup historie plateb.
- Automatické ukládání stavu při ukončení a znovunačtení při spuštění.
- A jistě další věci...

Problémy při implementaci

Nebyly žádné větší. Jen nemám zkušenosti s prací s konzolí. Takže jsem z počátku měl problémy se čtením vstupu, aby fungoval správně. Bylo potřeba vyřešit, aby čekání na vstup uživatele nebylo blokující.