2021

Tomoffice dokumentácia

OBJEKTOVO ORIENTOVANÉ PROGRAMOVANIE

TOMÁŠ KALNÝ

Tomáš Kalný
Streda 14.00

Zoznam využitých OOP princípov v projekte

- **Dedenie:** Z abstraktnej triedy *Pouzival* dedí *Klient* ale aj abstraktná trieda Zamestnanec z ktorej potom dedia zvyšní zamestnanci. Tiež mám dedenie z abstraktnej triedy *Tovar*z nej dedí *Zosit* a *TovarRozmer*.
- **Agregácia:** Sa nachádza vo viacerých častiach, každý používateľ má atribút login triedy *Login* a správy typu *Spravy, Klient* má *Adresu*, *Pracovník* stroja *zase Stroj*.
- **Polymorfizmus:** Prekonávam metódu toString() kvôli zobrazeniu informácií o prihlásenom používateľovi na domovskej obrazovke. *Pouzivatel* v metóde toString() vráti meno, email a adresu ako string oddelené novými riadkami. Klient túto metódu prekonáva pretože má aj adresu.
- **Preťažené metódy:** V triede AController mám preťaženú metódu zobraz_okno, ak je v argumente aj stage, prepnem celú scénu, ak je tam iba root a rozmery okna vyskočí nové okno navyše.
- **Oddelenie logiky od GUI:** Oddelenie realizujem tak, že si vytváram manuálne fxml súbory a k nim pripájam vlastné kontroléri, ktoré spracovávajú akcie a prepínanie medzi scénami. Každá scéna je 1 fxml súbor s vlastným controllerom.
- **Návrhový vzor observer:** Tento vzor využívam viac krát. Každú objednávku sleduje 1 singleton trieda *ManazerObjednavok*. Pri každej zmene objednávky (výroba nejakej časti tovaru) sa upozorní tento *ManazerObjednavok*, ktorý skontroluje či už nie je celá objednávka hotová a nemá sa posunúť skladníkovi na odoslanie. Tiež trieda Sklad má zoznam pozorovateľov (skladníkov) a keď treba sklad doplniť upozorní všetkých skladníkov. Posledný krát kedy využívam Observer je pri Správach používateľov vo výrobnom procese. Kde controller je vlastne observer pre *Spravy* a keď príde nová správa vie že ju má pripojiť do textArea. Tu observer implementuje rozhranie *PozorovatelSprav* ktoré definuje metodu notify s argumentom správy.
- **Návrhový vzor Strategy:** Vzor využívam pri pracovníkoch výroby a samotnom procese výroby. Mám interface *ObjSpracovanie*, ktorý implementuje každý pracovník a zároveň sladník. Tento interface definuje metódu spracuj_obj, ktorá má ako argument danú objednávku. Následne pri procese výroby volám túto metódu ((Vyroba)this.p).vyrob_tovar(o) kde sa vyberie „stratégia“ podľa toho či je to *PracovníkFotiek*, Zošitov alebo Obálok, prípadne keď je to skladník objednávka sa odošle a zmení sa aj text tlačidla, namiesto Vyrob je tam Odošli objednávku.
- **Multithreading:** Je využívaný pri výrobe. Výroba spočíva v tom, že pracovník skontroluje či je dostatok materiálu na sklade a ak hej zavolá metódu svojho stroja, ktorý vyrába daný produkt. Táto metóda (spusti_proces) vytvorí novú niť cez triedu *ProcesVyroby* (moja trieda ktorá dedí z Thread) do tohto procesu výroby si posúvam samotný stroj, objednávku a tovar už nie, vzhľadom na to, že je to vnorená trieda. Potom v *ProcesVyroby* v metóde run volám stroj.zacni_vyrabat. Táto metóda je synchronized a má na starosti výpis a samotnú výrobu tovaru. (Nachvíľu dám thread sleep aby tam bol nejaký časový rozdiel pri výrobe, že to chvíľu trvá).

- **Explicitné použitie RTTI:** Znova sa v projekte nachádza viac krát. Napr. viditeľnosť niektorých tlačidiel je určená podľa toho či je používateľ inštanciou Zamestnanca/Klienta/Vyroby atď. V ManazerObjednavok v metóde prirad_objednávku_pracovnikom podľa typu Tovaru priraďujem pracovníka, typ tovaru zisťujem cez instanceof. Tretie využitie RTTI je napr. vo funkcií vyrob_tovar() kde preskočím výrobu tovaru, ktorý nemá daný pracovník na starosti (pracovník fotiek vyrába iba fotky)
- **Lambda výraz:** Sa nachádza v triede *Spravy* v metóde toString() kde prechádzam všetky správy a postupne vytváram jeden dlhý string. Správy prechádzam cez foreach kde využívam lambda výraz.
- **Serializácia:** Pôvodne som mal serializáciu pri každom zatvorení celej aplikácie avšak od prezentovania som to zmenil na tlačidlo ulož (niekedy nechcem uložiť stav v akom je celá aplikácia). Serializujem používateľov, objednávky a sklad. Ak sa nepodarí deserializovať načítam používateľov a objednávky z textových súborov a sklad vytvorím nanovo.
- **Vnorená trieda:** Vnorená trieda ProcesVyroby sa nachádza v triede Stroj, ProcesVyroby dedí z Thread a vytvára novú niť, v ktorej sa nejaký čas (v závislosti od množstva tovaru) vyrába tovar cez stroj. Vzhľadom na to, že túto tried nikde inde v projekte nepotrebujem som sa rozhodol ju použiť ako vnorenú triedu.
- **Default metóda:** Je v rozhraní TonerSpotreba, ktoré obsahuje preťaženú default metódu vypocitaj_spotrebu_tonera, raz túto defaultnú metódu využívam pri vypočítaní spotreby tonera pre zošit, pričom pri fotkách je táto metóda prekonaná.

Poznámka: Čo sa zmien od prezentovania týka, pridal som vlastnú výnimku, upravil niektoré časti kódu (nefungoval napríklad správne Sklad). Tiež som odstránil interface prístup do skladu a pridal triedu Zamestnanec, ktorý vlastne plní podobnú funkciu, pretože zamestnancom prideľuje atribút sklad. Kód som aj okomentoval, pridal zobrazovanie vybavených objednávok, odosielanie objednávok a zmenil serializáciu ktorá je teraz vykonávaná kliknutím tlačidla ulož.

O projekte

Projekt sa sústredí na plánovanie a proces výroby papierových výrobkov, slúži klientom a zamestnancom firmy. V softvéri sa používateľ prihlási svojimi prihlasovacími údajmi a vykonáva operácie ktoré sú mu povolené. Môžu sa vytvárať nové objednávky (automaticky sa vypočíta cena objednávky a priradí konkrétnym pracovníkom), prezerať aktuálne objednávky a vybavené objednávky. Proces výroby spočíva vo vybratí konkrétnej objednávky, skontrolovania stavu skladu (ak nie je dostatok materiálu nemôže sa pokračovať s výrobou a sklad musí doplniť skladník), posunutie výroby danému stroju, ktorý určitý čas pracuje na výrobe, následne sa označí daný tovar ako vyrobený a čaká sa pokiaľ nie je vyrobený všetok tovar v objednávke. V prípade, že je v objednávke viacero

rôznych druhov tovaru (fotky, zošity, obálky), čaká sa kým, každý pracovník nevyrobí svoju časť. Potom ako je všetok tovar vyrobený sa objednávka automaticky označí ako pripravená na odoslanie a priradí sa skladníkovi. Skladník následne môže objednávku odoslať, čím sa označí ako vybavená. Po tomto procese, klient svoju objednávku vidí už v zozname vybavených objednávok. Taktiež existuje manažér, ktorý má prehľad o všetkých nevybavených a vybavených objednávkach, môže vyhodiť zamestnanca alebo prípadne prijať nového zamestnanca.

Zoznam dôležitých commitov

- https://github.com/OOP-FIIT/oop-2021-str-14-a-povazanova-to-0/commit/a62173c17362c6b51bee9c17c68012472777e13b
  a62173c17362c6b51bee9c17c68012472777e13b prvotný commit, kde som pushol veci, čo som mal dovtedy hotové, už tu som to mal celkom rozpracované
- https://github.com/OOP-FIIT/oop-2021-str-14-a-povazanova-to-0/commit/c11fbd459c0df16fb4a31301fa2ecde12ee94dbc
  c11fbd459c0df16fb4a31301fa2ecde12ee94dbc, ty som vlastne pridal proces výroby a spracovanie objednávok, priraďovanie pracovníkom, vytvárane nových objednávok, bola tu vytvorená celá trieda manažéra objednávok, multithreading, trieda stroja
- https://github.com/OOP-FIIT/oop-2021-str-14-a-povazanova-to-0/commit/1d8d0ed0a6e06500d9b5d8aa125ed8844093c97b
  1d8d0ed0a6e06500d9b5d8aa125ed8844093c97b pridal som sklad, doplňovanie jeho skladu, pridanie serializácie, preťaženia metód a pridanie lambda výrazov
- https://github.com/OOP-FIIT/oop-2021-str-14-a-povazanova-to-0/commit/31120e29338e19836c200c3191e8db770670f261 odosielanie objednávok, posunutie objednávok skladníkom, zobrazenie vybavených objednávok
- https://github.com/OOP-FIIT/oop-2021-str-14-a-povazanova-to-0/commit/86d6a2e9458c5f5ac33b89aa10edf564af684976 admin obrazovka, kde môže manažér manažovať zamestnancov (vyhodiť alebo pridať nového), úprava triedy ManazerObjednavok
- https://github.com/OOP-FIIT/oop-2021-str-14-a-povazanova-to-0/commit/b7e70eef611a922983222a53ddbf7d0445cfee76 presun triedy ProcesVyroby ako vnútornú triedu triedy Stroj

Tomáš Kalný
Streda 14.00

Poznámka:

Pre jednoduchšie testovanie pre Vás, v súbore používatelia, vidno login a heslo pre každého používateľa sú to 2 a 3 slovo v poradí. Tu poskytnem pár údajov pre rýchlejšiu kontrolu:

Každý používateľ má heslo 123

Zákazník: janko, Alzbeta

Manažér: anton

Skladník: jozo, silvia

Fotky: fero, fratisek

Zošit: zoro, zoltan

Obálky: oliver,olga