Programozási technológiák beadandó – dizájn dokumentum

Készítette: Molnár Adrienn

A beadandó feladat egy raktár menedzser program tervezése volt, amelynek a megadott funkcionalitások és alrendszerek megtervezett interfészeit és „csontvázát” kellett tartalmaznia. A feladatot az órán tanult tervezési minták segítségével kellett megoldani.

A projektemben egy fő raktár létezik és ez kommunikál a kettő alraktárral, amelyekben lego-t és társasjátékot tárolunk. A vevők úgy tudnak rendelni a termékek közül, hogy csak a fő raktárról tudnak. Amikor a fő raktárba termék érkezik, az értesíti az alraktárakat és a termék bekerül a megfelelő alraktárba. A vásárlók között megkülönböztetjük, hogy magánszemély vagy bolt adta-e le a rendelést. Mindkét vásárló típus esetében vizsgáljuk, hogy a vevőnek van-e elég pénze kifizetni a rendelést, nemleges válasz esetén a rendelés nem végrehajtható. A bolti vásárlás esetében mindkét termékkategóriánál van egy minimum határ, amit a rendelés végösszegének el kell érnie, ellenkező esetben nem hajtható végre a rendelés. Az Invoice osztály segítségével a rendelésekhez számla is kiállítható, amelyen szerepel a vevő neve, címe, a termékből vásárolt mennyiség, illetve a vásárolt termék. A programnak nincsen felhasználói felülete, így a tesztek bizonyítják a működést.

A program a következő osztályokat tartalmazza:

* MainStorage (A fő raktár. Implementálja az IObserverInfo interfészt)
* LegoStorage (Az egyik alraktár, a lego-kat tartalmazza, illetve implementálja az IObserver és az IProductOperations interfészeket.)
* BoardGameStorage (A másik alraktár, a társasjátékokat tartalmazza, illetve implementálja az IObserver és az IProductOperations interfészeket.)
* Product (A termék adatait tárolja.)
* Customer (A vásárlók adatait tárolja.)
* PersonCustomerStrategy (A magánszemélyekre vonatkozó rendelési stratégiát tartalmazza. Implementálja az ICustomerStategy interfészt.)
* ShopCustomerStrategy (A boltokra vonatkozó rendelési stratégiát tartalmazza. Implementálja az ICustomerStategy interfészt.)
* Order (A megrendeléseket tartalmazza.)
* Invoice (A számlához szükséges adatokat tartalmazza.)
* ProductSupplier (A beszállítók adatait tartalmazza.)
* CanNotPayException (Kivétel arra az esetre, ha a vásárlónak nincs elég pénze, a rendelés kifizetésére.)
* UnderMinCostException (Kivétel arra az esetre, ha a bolt túl alacsony végösszegű rendelést adott le.)

A program a következő interfészeket tartalmazza:

* ICustomerStrategy (A stratégia tervezési minta létrehozásához szükséges interfész. A PersonCustomerStrategy, illetve a ShopCustomerStrategy implementálja.)
* IObserverInfo (Az observer tervezési mintához szükséges interfész. A MainStorage implementálja.)
* IObserver (Az observer tervezési mintához szükséges interfész. A LegoStorage és a BoardGameStorage implementálja.)
* IProductOperations (A termékekkel kapcsolatos műveletekhez szükséges interfész. A LegoStorage és a BoardGameStorage implementálja.)

A program egy enum-ot is tartalmaz ProductTypeEnum néven, amely a termékek fajtáit tartalmazza.

A projekthez tartoznak teszt fájlok is, melyek a következők:

* CanNotPayExceptionTests
* UnderMinCostExceptionTests
* StorageTests
* OrderTests
* InvoiceTest

A projektben használt tervezési minták:

* Singleton
* Observer
* Stratégia

A singleton olyan programtervezési minta, amely egy objektumra korlátozza egy osztály létrehozható példányainak számát. A programban a fő raktár, azaz a MainStorage singleton-ként lett létrehozva, mivel csak egyre lesz belőle szükségünk.

Az observer egy olyan programtervezési minta, melyben egy objektum listát vezet alárendeltjeiről, és automatikusan értesíti őket bármilyen állapotváltozásról, többnyire valamely metódusuk meghívásán keresztül. A projektemben a fő raktár vezet listát az alraktárakról és ha termék érkezik a fő raktárba, az értesíti a két kisebb raktárat és berakja a terméket a megfelelőbe.

A stratégia tervezési minta lehetővé teszi, hogy egy algoritmus viselkedését a futás során válasszuk meg. Különböző viselkedéseket adhatunk meg hasonló objektumoknak. A programban a stratégia mintát a vevők megkülönböztetésére használtam aszerint, hogy magánszemély vagy bolt vásárol-e. A két vásárlói szerepkör annyiban tér el egymástól, hogy a boltoknak mindkét termékkategóriára van egy minimum végösszeg, ami alatt nem veszünk fel tőlük rendelést.



A program UML ábrája