GoF minták 2

- **Dependency Inversion**
  - Egy sor paraméterrel rendelkező konstruktornál használjunk interfészeket.
    - Dependency Injection
    - Inversion of Control -> IoC
    - Factory minták
- **Dependency Injection**
  - Lazán csatoltság kiterjesztéshez vezet a kódot, a kiterjeszthetőség pedig karbantarthatóvá.
  - Probléma -> A kód függjön absztrakcióktól, ne a konkrét implementációktól
  - Megoldás
    - Interfészt várjuk a konstruktorban paraméterként
    - Setter injektálás, amikor az objektumokat setter metódusok segítségével injektáljuk

- Factory method
  - Creational patlern
  - Új példány létrehozását leszármazott osztályokra bízzuk.
  - Probléma -> Objektumokat gyakran nehéz létrehozni és a konstruktorok nem flexibilisek
  - Megoldás -> Factory method-ok létrehozása, ahol létrehozzuk az új objektumokat, amivel vissza is térhetünk
  - Implementálás
    1. IFactory interfész létrehozása, tartalmaz egyéb műveleteket.
    2. Konkrét osztályok létrehozása és IFactory implementálása.
    3. Absztrakt Creator osztályok létrehozása egy üres IFactory Factory metódussal.
    4. Konkrét creator osztályok létrehozása és Creator osztály implementálása.
    5. Factory metódus implementálása, ahol adjuk vissza magát a konkrét product osztályt.
  - Előnyök és hátrányok
    - Előnyök -> Single responsibility és Open/closed elv
    - Hátrányok -> Sok alosztály miatt bonyolultá válhat a kód

- **Abstract Factory**
  - Több factory metódus "összefogása"
  - Ős factory tulajdonképpen
  - Probléma -> Különböző" feltételek alapján végrehajtani egy factory-t
  - Megoldás -> Dictionary vagy reflexióval
  - Implementálás
    1. Létrehozunk egy IFactory interfészt.
    2. A különböző" leszármazott factory osztályokban implementáljuk az IFactory interfészt.
    3. Létrehozzuk magát az Abstract Factory osztályt.
       - Dictionary<IFactory, string> privát
       - Konstruktorban inicializáljuk és hívjuk meg a Factory metódusokat
       - Legyen egy metódus, ahol kulcs alapján lefut a megfelelő" Factory metódus
  - Előnyök és hátrányok:
    - Előnyök -> Elrejti a konkrét osztályokat
    - Hátrányok -> Dictionary-t bővíteni kell

- **IoC minták**
  - Dependency Injection
  - Observer pattern
  - Template method

• Obszerver

- Behavioral
- Probléma
  - Potenciális vevő mindennap elmegy a boltba megkérdezni, hogy van-e XY márkájú telefonból új kiadás.
  - Bolt minden új kiadásról küld emailt, így az spam lehet.
  - Mindkét fél pazarolja a saját idejét.
- Megoldás
  - Eseményre való feliratkozás

- Implementálás
  1. IObserver interfész létrehozása egy StateChange(ISubject) metódussal, ami felel a frissítésért.
  2. ISubject interfész létrehozása Subscribe(IObserver); UnSubscribe(IObserver) és Notify() metódussal.
  3. RealSubject osztály létrehozása, ahol implementáljuk az ISubject interfészt és metódusait.
     - State privát mező, lehet int, string, stb...
     - IObserver lista létrehozása
     - Subscribe metódusban hozzáadjuk az IObserver paramétert.
     - UnSubscribe metódusban kitöröljük az IObserver paramétert.
     - Notify metódusban bejárjuk a listát és meghívjuk az aktuális elem StateChange metódusát. (Egy saját metódusban hívjuk meg a Notify() metódust)
  4. Konkrét osztályban implementáljuk az IObserver interfészt, ahol ha a State egyenlő valamivel, akkor végrehajtunk egy saját logicot.

- Előnye -> Open/Closed elv
- Hátrány -> A subscribereket véletlenszerű sorrendben értesíti

- Template method
  - Nagyobb metódusok felbontása kisebb lépésekre.
  - Probléma
    - Van egy metódusunk, ami felel a fájlok beolvasásáért.
    - PDF, CSV, XML fájlok beolvasása ugyanaz, csak a parsolás logikája más
  - Megoldás
    - Fájlbeolvasó metódus
    - Parse metódus
  - Implementálás
    1. ITemplate interfész egy TemplaMethod() metódussal.
    2. Template osztály létrehozása és ITemplate interfész implementálása.
       - TemplatMethod()-ban hívjuk meg a konkrét osztályokban lévő kisebb lépéseket. (Ezekre referenciaként hivatkozni kell)
  - Előnyök
    - Kódduplikációt elkerüljük
    - Átláthatóbb a kisebb lépések miatt
  - Hátrányok
    - TemplateMethod()-ból "god" object jöhet létre, amit egy idő után nehéz karbantartani a sok kisebb lépések miatt