GoF alapelvek (1)

- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.
- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Programtervezési minták: Újrahasznosítható elemek objektumközpontú programokhoz*. Kiskapu, 2004.

GoF alapelvek (2)

- **Interfészre programozzunk, ne implementációra!**
  - *„Program to an interface, not an implementation.”*
- Lásd a létrehozási mintákat!

GoF alapelvek (3)

- **Részesítsük előnyben az objektum-összetételt az öröklődéssel szemben!**
  - *„Favor object composition over class inheritance.”*
- A két leggyakoribb módszer az újrafelhasználásra az objektumorientált rendszerekben:
  - Öröklődés (fehér dobozos újrafelhasználás)
  - Objektum-összetétel (fekete dobozos újrafelhasználás)
- A fehér/fekete dobozos jelző a láthatóságra utal.

GoF alapelvek (4)

- Az öröklődés előnyei:
  - Statikusan, fordítási időben történik, és használata egyszerű, mivel a programozási nyelv közvetlenül támogatja.
  - Az öröklődés továbbá könnyebbé teszi az újrafelhasznált megvalósítás módosítását is. Ha egy alosztály felülírja a műveletek némelyikét, de nem mindet, akkor a leszármazottak műveleteit is megváltoztathatja, feltételezve, hogy azok a felülírt műveleteket hívják.

GoF alapelvek (5)

- Objektum összetételt használó tervezési minták:
  - Szerkezeti objektumminták: (objektum) illesztő, híd, összetétel, díszítő, homlokzat, pehelysúlyú, helyettes.
  - Viselkedési objektumminták: felelősséglánc, parancs, bejáró, közvetítő, emlékeztető, megfigyelő, állapot, stratégia, látogató.

GoF alapelvek (6)

- Az öröklődés hátrányai:
  - Először is, a szülőosztályoktól örökölt megvalósításokat futásidőben nem változtathatjuk meg, mivel az öröklődés már fordításkor eldől.
  - Másodszor – és ez általában rosszabb –, a szülőosztályok gyakran alosztályaik fizikai ábrázolását is meghatározzák, legalább részben. Mivel az öröklődés betekintést enged egy alosztálynak a szülője megvalósításába, gyakran mondják, hogy az öröklődés megszegi az egységbe zárás szabályát. Az alosztály megvalósítása annyira kötődik a szülőosztály megvalósításához, hogy a szülő megvalósításában a legkisebb változtatás is az alosztály változását vonja maga után.
  - Az implementációs függőségek gondot okozhatnak az alosztályok újrafelhasználásánál. Ha az örökölt megvalósítás bármely szempontból nem felel meg az új feladatnak, arra kényszerülünk, hogy újraírjuk, vagy valami megfelelőbbel helyettesítsük a szülőosztályt. Ez a függőség korlátozza a rugalmasságot, és végül az újrafelhasználhatóságot.

GoF alapelvek (7)

- Az objektum-összetétel dinamikusan, futásidőben történik, olyan objektumokon keresztül, amelyek hivatkozásokat szereznek más objektumokra.
- Az összetételhez szükséges, hogy az objektumok figyelembe vegyék egymás interfészét, amihez gondosan megtervezett interfészek kellenek, amelyek lehetővé teszik, hogy az objektumokat sok másikkal együtt használjuk.

GoF alapelvek (8)

- Az objektum összetétel előnyei:
  - Mivel az objektumokat csak az interfészükön keresztül érhetjük el, nem szegjük meg az egységbe zárás elvét.
  - Bármely objektumot lecserélhetünk egy másikra futásidőben, amíg a típusaik egyeznek.
  - Továbbá, mivel az objektumok megvalósítása interfészek segítségével épül fel, sokkal kevesebb lesz a megvalósítási függőség.
  - Az öröklődéssel szemben segít az osztályok egységbe zárásában és abban, hogy azok egy feladatra összpontosíthassanak.
  - Az osztályok és osztályhierarchiák kicsik maradnak, és kevésbé valószínű, hogy kezelhetetlen szörnyekké duzzadnak.

GoF (9)

- Másrészről az objektum-összetételen alapuló tervezés alkalmazása során több objektumunk lesz (még ha osztályunk kevesebb is), és a rendszer viselkedése ezek kapcsolataitól függ majd, nem pedig egyetlen osztály határozza meg.