Öröklődés

Az előző órán már találkoztunk az objektumorientáltság egyik alapfogalmával, az egységbezárással (*Encapsulation*), mely szerint az adatokat és a rajtuk végzett műveleteket egységbe zártuk, egy osztályba. most pedig megismerkedünk az öröklődéssel.

Az öröklés osztályok között valósul meg, egy szülő (*ős, base, parent*) és gyerek (*leszármazott, derived*) között.

- **ős** osztály: az az osztály, amelytől egy másik osztály örököl.
- **leszármazott** osztály: az az osztály, amelyik egy másik osztálytól örököl.
- osztály- vagy **objektumhierarchia**: az osztályok között az öröklési viszonyok meghatározásának következtében kialakuló (család) faszerkezet.

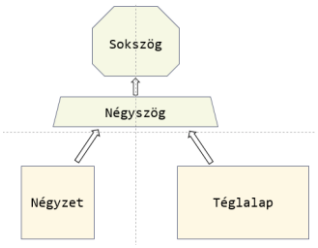
Ennek során a gyerekosztály örökli a szülőjének tulajdonságait és viselkedését. Ez jó dolog, mert nem kell újraírnunk őket, viszont a szülőtől örökölt egyes metódusokat speciálisabban is megvalósíthatjuk, felülírhatjuk őket(*override*), vagy akár újakat is definiálhatunk.

Javaban csak egyszeres öröklődés van!

Ez azt jelenti, hogy egy osztálynak nem lehet kettő, vagy több szülőosztálya. Azt viszont nem zárja ki, hogy egy osztálynak több gyerekosztálya legyen, vagy, hogy a gyerekosztálynak lehessenek saját gyerekosztályai.

`Az Objektum Orientált felfogás és a Java Programozás egyik nagy előnye az úgy-nevezett Öröklés.`

Az Öröklés azt jelenti, hogy egy bizonyos hierarchia alakul ki az Objektumok között. Az öröklésben mindig két végpont van, az ***Ős*** és az azt megöröklő ***Gyerek***. A Gyerek megörökli az Ős minden tulajdonságát és tovább bővítheti azokat. Erre a gondolatra egy példa a Négyszög és a Téglalap. *Minden Téglalap Négyszög, de nem minden Négyszög Téglalap.*



Ugyanúgy, mint a valóságban, így az Objektumok között is lehet definiálni ilyen jellegű kapcsolatot, sőt kapcsolati láncokat, ugyanis a Gyerek lehet Őse egy másik Gyereknek is. Sőt egy Ősnek több Gyereke is lehet

Öröklés előnyei

Ezzel a felfogással nagyon jól szét lehet választani a szerepköröket az Objektumok között, szét lehet osztani a felelősséget. Azonban a szabályokat is nagyon jól meg tudom kötni, hiszen már egészen magas szinten meg tudom mondani, hogy mik azok a tulajdonságok vagy műveletek, amik az öröklési hierarchia minden tagjában meg kell jelenjenek.

A legnagyobb előnye mégis az általánosítás, hiszen ezentúl nem kell pontosan megmondanom, hogy melyik Objektumra van szükségem, elég, ha megjelölöm a hierarchia tetején álló Objektum valamelyikét. Például ha egy *Négyszög* osztálybeli Objektumot várok, akkor kaphatok *Négyzetet* vagy *Téglalapot*, mind a kettő Négyszög, tehát mind a kettő megfelelő referencia.

Ez a megközelítés a Projekt munkák során rendkívül elterjedt, gyakran használt. Sőt, magában a Java nyelvben is rengeteg olyan elem van, amely kihasználja az öröklési láncok nyújtotta előnyöket. Érdemes tisztában lenni az alapokkal, mert egy komplexebb problémában gyorsan a segítségünkre lehet, ezzel időt és pénzt spórolva.

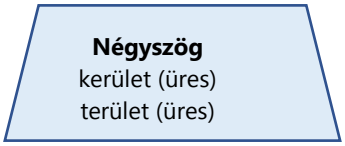
Öröklés a Java nyelvben

Maradva a korábbi példánál Négyszögek Kerületét és Területét akarom kiszámolni. Egészen pontosan a Négyzet és a Téglalap kerületére és területére vagyok kíváncsi. Valamint akarok egy külön osztályt, aminek a feladata a Négyszögek kerületének és területének a megjelenítése.

- Négyzet: kerület, terület, *beleírható kör* ← a leszármazott új tulajdonsága
- Téglalap: kerület, terület
- KI: kerület, terület

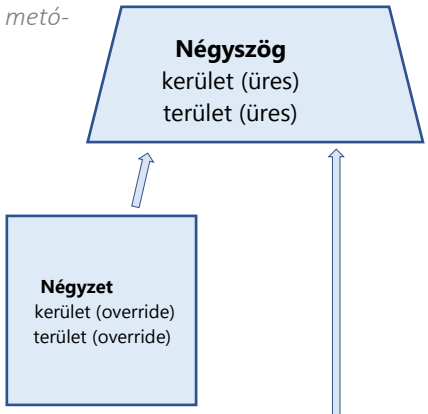
Definiálok tehát egy Ős osztályt, aki jelen példában nem egy igazi osztály, csupán az a célja, hogy közös Ősként szolgáljon a Gyerek osztályoknak. *Az ilyen osztályokat a Java-ban* ***Abstract*** *kulcsszóval jelöljük, és nem lehet őket példányosítani.* Ebben az osztályban definiálom a szükséges függvényeket.

```
package négyszögöröklés;
abstract class Négyszög {
/* Az abstract osztályban a metódusokat csak definiálni lehet, valódi implementációt
majd a Gyerek osztályban kapnak */
   public abstract int kerület();
   public abstract int terület();
}
```

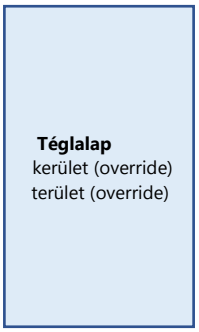


A következő lépések a gyerek osztályok megvalósítása. `New -> Java Class:` **`Négyzet`**

```
package négyszögöröklés;
/* A Négyzet osztályba az a oldal hossza kell, valamint a kerület() és a terület() metódusok az Ős (abstract) osztályból */
public class Négyzet extends Négyszög {
   private int a;
   public Négyzet(int a) {
      this.a = a;
   }
   public int kerület() {
      return a * 4;
   }
   public int terület() {
      return a * a;
   }
}
```




```
package négyszögöröklés;
/* A Téglalap osztályba az a,b oldal hossza kell, valamint a kerület() és a terület() metódusok az
   Ős (abstract) osztályból */
public class Téglalap extends Négyszög {
   private int a, b;
   public Téglalap(int a, int b) {
      this.a = a;
      this.b = b;
   }
   public int kerület() {
      return 2*(a + b);
   }
   public int terület() {
      return a * b;
   }
}
```




```
package négyszögöröklés; // ez már csak hab a tortán
/* Definiáljuk a megjelenítő, kiíratő osztályt. Ennek egyetlen metódusa lesz, ami paraméterül vár egy Négyszög osztályt.
Mivel tudjuk, hogy a Négyszög osztály ismeri a kerület- és területszámítási metódusokat, ezért ezeket már fel tudjuk használni
a megjelenítéshez. */
public class Printer {
   public void PrintKerületTerület(Négyszög négyszög) {
      System.out.println("kerület: " + négyszög.kerület() + "\nterület: " + négyszög.terület()+"\n");
   }
}
```

Nézzük meg, hogyan néz ki most a NégyszögÖröklés osztályunk:

```
package négyszögöröklés;

public class NégyszögÖröklés {
   private static Négyzet négyzet;
   private static Téglalap téglalap;
   private static void f1() {
  // először létrehozok egy-egy példányt a két Gyerek osztályból
      négyzet = new Négyzet(5);
      téglalap = new Téglalap(5, 10);
   }
   private static void f2() {
  // egy példányt a megjelenítő osztályból
      Printer printer = new Printer();
      printer.PrintKerületTerület(négyzet);
      printer.PrintKerületTerület(téglalap);
   }
   public static void main(String[] args) {
      f1();
      f2();
   }
}
```

Először létrehozok egy-egy példányt a két Gyerek osztályból valamint egy példányt a megjelenítő osztályból is. Ezután meghívom a megjelenítő osztály metódusát a megfelelő Gyerek osztályokkal. ***Mivel minden Négyzet és minden Téglalap egyben Négyszög is, ezért a megjelenítő osztály elfogadja ezeket a bemeneti értékeket.***

Ez a példa úgy gondolom jól mutatja be az öröklés előnyét.

Ha az öröklést most nem tudtam volna kihasználni, akkor két külön kiíratómetódust kellett volna definiálnom a Megjelenítő osztályban, egyet a Négyzetre, egyet pedig a Téglalapra.

Ez egy ekkora példában szemmel láthatóan nem lett volna akkora plusz munka, de egy nagyobb Projektben már komoly gondot tudna okozni.