Programozási nyelvek Java

3. gyakorlat

Néhány alapvető fogalom

- **Hatókör**: Ahol a változó használható
- **Élettartam**: Ahol a változó még "*él*"
- **Láthatósági módosítószavak**:
  - public: minden más osztályból elérhető (függvény, adattag, vagy osztály)
  - protected: adott csomagon belülről, valamint a származtatott osztályokból elérhető
  - private: kizárólag az adott osztályon belülről használható
  - (default): ha nem deklarálsz semmit, akkor ez az alapértelmezett, ún. *package-private* láthatóság (csak az adott csomagon belül látszik)
- **Adatelrejtés**: A reprezentáció elrejtése (specifikáció, reprezentáció, implementáció szétválasztásának támogatása), csak a publikus metódusokon (interfészen) keresztüli elérés támogatása

Objektum

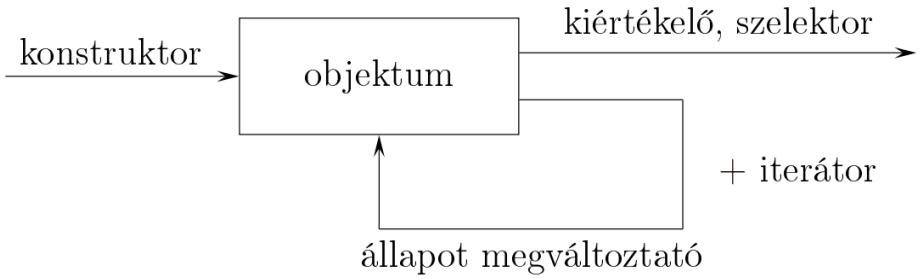
- Attribútumok és műveletek tartoznak hozzájuk
- Állapot tartozik hozzá (az attribútumok konkrét értékei határozzák meg)
- Korlátolt láthatósággal rendelkezik
  - Látható rész (deklaráció) = interfész, azaz az export és az import műveletek formái és jelentései
  - Láthatatlan rész (reprezentáció) = objektum ábrázolása, és a szolgáltatásainak a megvalósítása
- Minden objektum egy osztály példánya
- Speciális művelet (a konstruktor) és a new kulcsszó segítségével hozható létre
- Az osztályobjektumok referenciákon keresztül érhetők el (memóriacímet tárolnak)
  A referenciák értéke lehet null. (pl: Person somebody;)
- Felszabadítással nem kell foglalkozni (GC miatt)
- final kulcsszó: a referencia nem módosítható, de a mutatott érték megváltozhat

Objektum műveletei

- **Export műveltek:** azok a műveletek, amelyeket más objektumok végezhetnek rajta

- **Import műveletek:** azok a műveletek, amelyeket az objektum igényel ahhoz, hogy az export szolgáltatásait nyújtani tudja

Az export műveletek csoportjai



- **Konstruktor műveletek:** az objektum létrehozására, felépítésére szolgálnak
- **Kiértékelő műveletek:** az objektum bizonyos jellemzőit kérdezik le
- **Szelektor műveletek:** az objektum bizonyos részét kiemelik
- **Állapot megváltoztató műveletek:** az objektum attribútumainak az értékét változtatják meg
- **Iterátorok:** az objektum felépítésében részt vevő komponensek bejárására szolgáló eljárások

Osztály

- Hasonló tulajdonságú objektumok egy halmaza
- Az osztálynak lehetnek attribútumai, paraméterei, műveletei
- Speciális művelet a konstruktor
    - Eze(ke)n keresztül hozható létre ilyen típusú objektum
    - Neve megegyezik az osztály nevével
    - Lehet paramétere is
- Tartozhat hozzá még import felület is
- Az osztályokhoz is definiálható láthatóság
- Lehet konkrét vagy absztrakt
- Lehet paraméteres osztály (sablon): közös formával rendelkező osztályok egy osztályát definiálja, olyan formális paraméterekkel, amelyeknek sem a típusuk, sem a korlátozásuk nincs megadva
- Konvenció: A névben szereplő összes szó első betűje nagybetű

Egy osztályimplementáció

```
public class Kerekpar
{
    private int    azonosito;
    private String tipus;
    private Color  szin;

    public Kerekpar()   { ... }
    public void kolcsonzes(Szemely ki)  { ... }
    public void javitas()      { ... }
}
```

Egy osztály adattagjaihoz, amelyeket nem csak az adott osztályban szeretnénk felhasználni, írni kell egy megfelelő lekérdező (getter) és egy beállító (setter) metódust (az adatelrejtés elve miatt).

```
get<Változó neve>()
set<Változó neve>(<új érték>)
```

Több osztály használata, csomagok

Legyen a Hello.java fájl tartalma a következő:

```
package hello;

public class Hello
{
        public String Hello(final String name)
        { return "Hello " + name + "!"; }
}
```

Készítsünk hozzá egy tesztelő osztályt, amiben szeretnénk meghívni a hello csomagban található Hello osztály Hello függvényét.

```
import hello.*;

/**
 * @author Dorian Batha
 * @version 2010.09.22.
 */
public class Tester
{
        /**
         * Nincsenek argumentumok!
         * @param args
         */
        public static void main(String[] args)
        {

                Hello h = new Hello();
                System.out.println(h.Hello("Dorian"));
        }
}
```

Az fájl elején található import kulcsszóval tudok importálni. A * azt jelenti, hogy a hello csomag minden elemét szeretném importálni. Ekkor már tudom használni a Hello osztályt. Ezután létrehozok egy példányt a Hello osztályból a new kulcsszóval. Ezek után már a csak meg kell hívnom a Hello függvényt a megfelelő paraméterrel és visszaadja az eredményt, amit kiíratok a konzolra.

Fordítás

Ha létrehozzuk a csomagoknak megfelelő könyvtárszerkezetet, akkor elég a fő fájlt lefordítani és (ha megfelelő a könyvtárszerkezet) megtalálja az importált csomag osztályait és lefordítja magának.

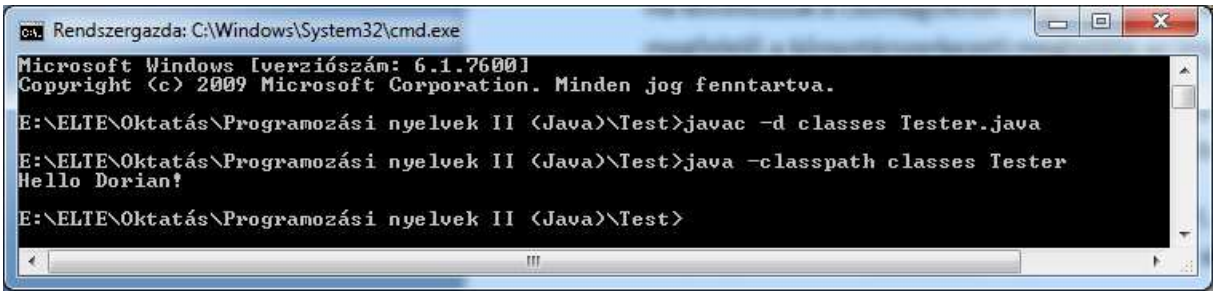
Tehát a Hello.java fájl-t a hello csomagban található (package hello;). Így létre kell hozni egy hello (ami a csomag neve) könyvtárat és benne kell elhelyezni a Hello.java fájl-t.

Miután elkészült a megfelelő könyvtárszerkezet, fordítsuk le a fő fájlunkat (Tester.java) és a fordító meg fogja találni a Hello.java fájl-t és lefordítja azt is.

A –d kapcsolóval jelezhetjük a fordítónak, hogy a class fájlokat a classes könyvtárba tegye bele (a classes könyvtárat létre kell hozni).

Futtatás

Végül pedig futtassuk a Tester-t. A –classpath kapcsolóval adom meg, hogy a lefordított objektumokat a classes könyvtárban találja.


Rendszergazda: C:\Windows\System32\cmd.exe
Microsoft Windows [verziószám: 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Minden jog fenntartva.
E:\ELTE\Oktatás\Programozási nyelvek II (Java)\Test>javac -d classes Tester.java
E:\ELTE\Oktatás\Programozási nyelvek II (Java)\Test>java -classpath classes Tester
Hello Dorian!
E:\ELTE\Oktatás\Programozási nyelvek II (Java)\Test>

Feladatok

1. Készítsétek el a honlapon található Hello programhoz a megfelelő könyvtárszerkezetet, majd fordítsátok és futtassátok a programot.
2. Valósítsatok meg egy **Calculator** nevű osztályt és rakjátok bele a **calculator** csomagba. Az osztálynak legyen egy **Add** (összeadás) és egy **Mul** (szorzás) nevű művelete, amelyek két egész számot várnak paraméterül és visszaadják az eredményt.
   Ezután futtasd le a honlapon található CalculatorTester.java tesztfájlt
3. Készítsetek egy új osztályt **Coordinate** névvel, és rakjátok a **coordinates** csomagba. Ez az osztály reprezentáljon egy kétdimenziós egész értékű koordinátát. írjátok meg a getter, setter műveleteket (getX, setX, getY, setY).
4. Készítsünk még egy osztályt **CoordinateOperations** névvel, szintén a **coordinates** csomagba. Írjunk az osztályba egy függvényt **MirrorO** névvel, ami a paraméterként megadott koordinátát tükrözi az Origóra és visszaadja az eredményt.
   Tesztelésképp használjátok a honlapon található CoordinateTester.java fájlt.
5. Az előző osztályba kerüljön bele egy újabb függvény **Distance** névvel, ami kiszámolja a paraméterben megadott két koordináta távolságát.
   A tesztfájlt egészítsétek ki a Distance függvény tesztelésével.
6. Készítsetek egy **Vector** nevű osztályt szintén a **coordinates** csomagba, ami egy kétdimenziós egész vektort reprezentál. Ezután egészítsétek ki a CoordinateOperations osztályt egy **Translate** művelettel, ami vár egy koordinátát és egy vektort paraméterül és visszaadja a koordináta eltoltját a vektorral.
   A tesztfájlt egészítsétek ki a Translate függvény tesztelésével.