Adatbázis és szoftverfejlesztés elmélet

Témakör 7.

Összefoglalás

1. Objektum adattípus

A programozás technikájának fejlődése olyan, hogy a programozást minden módon igyekszik a hétköznapi gondolkodáshoz közelíteni. Ennek a fejlődésnek az eredménye az objektumok megjelenése a programozási nyelvek újabb generációiban, valamint a tisztán objektum alapú nyelvek megjelenése. Az objektumokat először a Smalltalk programozási nyelvben, majd a Turbo Pascalban, illetve később a C nyelv bővítéseként megvalósult C++ nyelvben vezették be.

Az objektumok megjelenése hatott a programozás módszereire is. Manapság már majdnem minden modern programozási nyelv tartalmazza az objektumorientált programozás (= Object Oriented Programming => OOP) elemeit (pl.: Delphi, PHP vagy JavaScript). Teljes egészében objektumorientált nyelvek például: Java és C#.

Objektumok

A valós világban létező dolgok tulajdonságokkal jellemezhetők. A valós világ dolgait általánosságban az angol nyelv object-nek, azaz objektumoknak hívja. A programozásban a valós világ egyes dolgainak tehát objektumokat feleltethetünk meg, az objektumokat pedig tulajdonságaikkal (tulajdonság = property) jellemezzük. Ha egy objektum megváltozik, az azt jelenti, hogy egy vagy több tulajdonsága megváltozik.

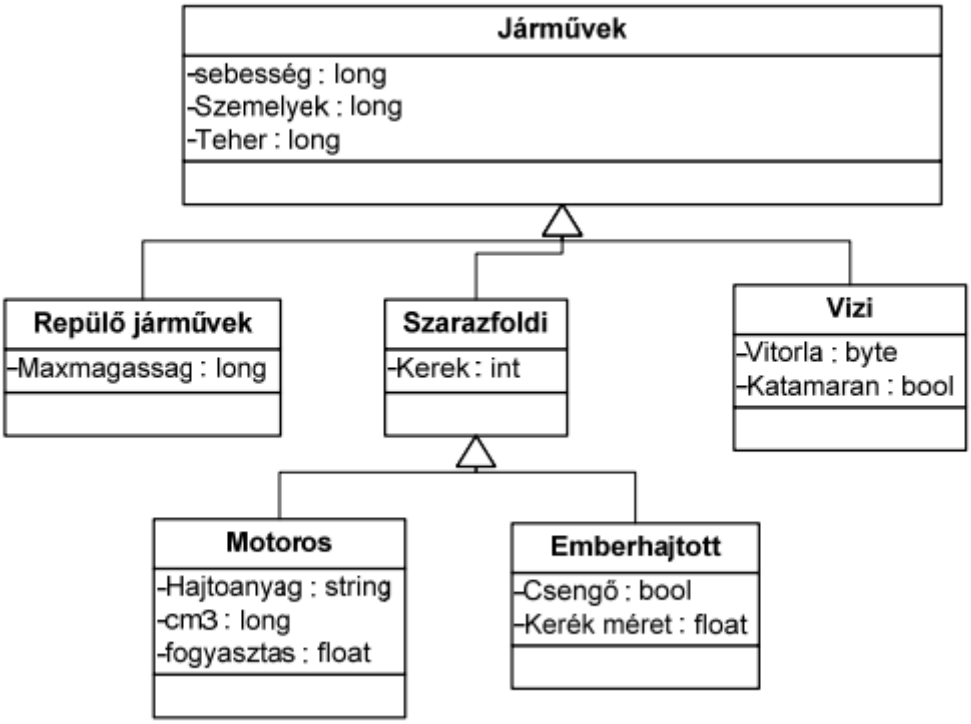
Üzenetek

Mitől változik meg egy objektum állapota. A külvilágból ingerek érik – ezeket általánosságban üzeneteknek (üzenet= message) hívjuk – és az objektum az ingerekre reagál, kijelzi, törli, módosítja valamelyik tulajdonságát, állapotát, és üzenetet küldhet más objektumnak. Az objektumokban **metódusoknak** hívjuk az üzenetet küldő, fogadó és azokra reagáló eljárásokat.

2. Osztályok az objektumok csoportjai

Az objektumokat tulajdonságaik alapján csoportokba, **osztályokba** sorolhatjuk. Például a járműveknek vannak olyan tulajdonságaik, amelyek alapján megalkothatjuk a járművek osztályát. Járműnek nevezzük azt, ami közlekedik, szállít, stb. A járműveknek vannak olyan tulajdonságai, amelyek minden járműre igazak, például a szállítható személyek száma, a szállítható teher mennyisége, a maximális sebessége, stb., Ugyanakkor megalkothatjuk az osztályok specializált eseteit is, mint például a földön közlekedő járművek, a repülő járművek vagy a vízen közlekedő járművek.

Amikor egy osztályba tartozó objektumok egy csoportja egy vagy több plusz tulajdonság alapján megkülönböztethető az osztály többi tagjától, akkor e tagokat **részosztályba** sorolhatjuk, amelynek a tagjai minden tulajdonsággal rendelkeznek, amivel a fő osztály tagjai rendelkeznek, de további speciális tulajdonságaik is vannak. Az alábbi ábra a járművek osztályt és részosztályainak egy lehetséges vázlatát mutatja:



A fő osztály a *Járművek* nevű osztály. A *Repülő járművek*, a S*zárazföldi* és a *Vízi* járművek a Járművek osztály leszármazottjai. A leszármazott osztályok minden tulajdonsággal rendelkeznek, amivel az ősük, de további tulajdonságok is érvényesek rájuk.

A valós világ modellezése

A programok a valós világot modellezik. Amikor a valós világot modellezni akarjuk, akkor érdemes a világ objektumaiból csoportokat alkotni közös jellemzőik alapján. Ha egy csoport egyes tagjainak vannak speciális tulajdonságai, akkor azokból érdemes egy leszármazott csoportot alkotni.

Definíció:

*A programokban a valós világ objektumainak csoportjait **osztályok**nak (=class) nevezzük. Az osztály absztrakt fogalom. Olyan összetett adattípus, amelyet a programozó meglévő elemi vagy más összetett adattípusokból épít fel.*

*Az osztályban lévő összetevő adattípusokat az osztály **tulajdonságainak** hívjuk.*

*Az osztályok tartalmazzák azt a programkódot is, amelyek segítségével az osztály tagjai tudnak reagálni a külső környezetből érkező üzenetekre. Ez a programkód formailag függvényekből és eljárásokból áll. Az így definiált eljárásokat és függvényeket **metódusoknak** hívjuk.*

Röviden tehát az osztály mindig egy adattípust jelöl, a valóság objektumainak egy csoportját. Ha ennek az osztálynak egy konkrét objektumát akarjuk kezelni, akkor létre kell hozni az osztály egy példányát.

Definíció:

***Példányosítás**nak nevezzük, amikor egy osztály konkrét előfordulását létrehozzuk. Az így létrejött programegységet **objektumnak** (=Object) hívjuk. A programokban ez egy változó, aminek típusa az osztály, amiből példányosítottuk. Az **objektumok** olyan zárt programozási egységek, amelyek az kezelni*

kívánt adatokon kívül tartalmazzák azokat az eljárásokat és függvényeket is, amelyek az objektumok megfelelő kezelésére képesek.

3. Objektumorientáltság alapelvei

Az objektumorientált programozás – az objektumok használata kissé más gondolkozást kíván meg a programozóktól, mint a hagyományos procedurális programozás. A program tervezése áttolódik az objektumok hierarchiájának átgondolt megtervezésére, továbbá az objektumok részeinek megfelelő kód létrehozására. A legtöbb fejlesztőrendszerben már létezik az objektumoknak egy olyan hierarchiája, amely a képernyőkezelésnél, az állományok elérésénél vagy a felhasználói felület kialakításánál elengedhetetlen.

Hogy egy programozási egységet objektumoknak tekintsünk az alábbiakban tárgyalt három alapvető tulajdonság megléte szükséges.

3.1. Egységbezárás

Az objektumoknak azt a tulajdonságát, hogy az adatmezőkön kívül az őt kezelő eljárások vagy függvények is az adattípus részét képezik, **egységbezárásnak** (**encapsulation**) hívják. Az objektum adatait manipuláló függvényeket, eljárásokat **metódusoknak** hívjuk.

Speciális metódusok

Az objektumoknak lehetnek speciális metódusai:

A **konstruktor** metódus akkor fut le, amikor egy objektumot, futás közben létrehozunk. A konstruktor biztosítja a helyet a memóriában az adatok és az egyéb kódok részére.

A **destruktor** metódus akkor fut le, amikor az objektum megszűnik. Annyi a feladata, hogy a megfelelő memóriaterületeket felszabadítja, a megnyitott fájlokat lezárja. A fejlesztőrendszerek automatikusan létrehozzák az egyszerűbb objektumok konstruktorjának és destruktorjának kódját, a programozónak esetleg hivatkozni sem kell rájuk.

3.2. Öröklődés

Az objektumorientált programozás egyik leghasznosabb része az öröklődés. Az öröklődés során származtatással hozunk létre egy már meglévő osztályból egy újat (leszármazott osztály v. gyermekosztály), amely rendelkezni fog az eredeti osztály (ősosztály v. szülő osztály) minden tulajdonságával, és nekünk csak a kívánt plusz funkciókat kell megvalósítanunk. A leszármazott osztályból példányosított objektumok öröklik őseik tulajdonságait, azaz minden eljárást, metódust és adatmezőt, amivel azok rendelkeznek. Ennek megfelelően a leszármazott objektumban is használhatjuk az ősobjektum metódusait.

3.3. Többalakúság (Polimorfizmus)

Ha egy leszármazott objektum metódusát ugyanolyan néven definiáljuk (felülírjuk), mint egy ősének metódusát, akkor a program futásakor fizikailag más eljárást kell használnia a rendszernek – az éppen használt objektumtól függően. Mindig az aktuális adattípusnak megfelelő metódus lesz végrehajtva. Például a **rajzol()** nevű metódus nem lehet ugyanaz pont, vonal, kör és sokszög esetén. A polimorfizmus azt eredményezi, hogy a program a megfelelő

metóduspéldányt választja ki futás közben a rendelkezésre álló ugyanolyan nevű metódusok közül.

Kérdések:

1. Mi a szerepe az objektumoknak és az üzeneteknek a programozásban?
2. Ismertesse az osztály (class) definicióját!
3. Mik az a metódusok?
4. Mi az objektum és a példányosítás?
5. Mit nevezünk egységbezárásnak?
6. Milyen speciális metódusok lehetnek egy objektumban és mikor hajtódnak végre?
7. Milyen kapcsolatot jelent az öröklődés az objektumok között?
8. Ismertesse a többalakúság jelentését!