



BME
Budapesti Műszaki és Gazdaságtudományi Egyetem

KJIT
Közlekedés- és Járműirányítási Tanszék

Programozás

BMEKOKAA146

Dr. Bécsi Tamás

7. előadás

AZ OOP alapelvei

- **Egységbezárás**(Encapsulation)
 - Az adatokat es a hozzájuk tartozó eljárásokat egyetlen egységben (osztályban) kezeljük. Az osztály adatmezői tárolják az adatokat, a metódusok kommunikálnak a külvilággal.
- **Öröklés**(Inheritance)
 - Az osztály továbbfejlesztése. Ennek során a származtatott osztály örökli az őosztálytól az összes attribútumot, metódust. Ezeket azonban újakkal is kibővíthetjük, ill. bizonyos szabályok mellett az örökölt metódusokat is megváltoztathatjuk.
- **Sokalakúság**(Polymorphism)
 - Ugyanarra a metódusra a különböző objektumok különbözőképpen reagáljanak. A származtatás során az őosztályok metódusai képesek legyenek az új, átdefiniált metódusok használatára újraírás nélkül is.
 - Ezt virtuális (vagy dinamikus) metódusokkal érhetjük el.

Osztályok

```
class osztálynév [: szülő, ...]
{
  • Osztálytaglista
    • Konstruktorok
    • Destruktorok
    • Adattagok
    • Metódusok
};
```

- Minden osztály egy közös ősből, az objectosztályból származik

Hozzáférési szintek osztályok esetében

1. A **private** tagokat csak az adott osztályon belülről érhetjük el.
2. Az osztályok publikus **public** mezőit bárhonnán elérhetjük, módosíthatjuk.
3. A **protected** mezők az osztályon kívüliek számára nem elérhetőek, míg az utódosztályból igen.
4. Az **internal** mezőket a készülő program osztályaiból érhetjük el.
5. A **protected internal** elérés valójában egy egyszerű vagy kapcsolattal megadott hozzáférési engedély. A mező elérhető a programon belülről, vagy az osztály utódosztályából! (Egy osztályból természetesen tudunk úgy utódosztályt származtatni, hogy ez nem tartozik az eredeti programhoz.)

Egyszerű példa

Adattagok

```
class Ember
{
    private int életkor;
    public string nev;
}
```

.....

```
Ember e = new Ember(); //automatikus konstruktor
e.nev = "Béla"; //működik
e.életkor= 12; //nem működik
```

Egyszerű példa

Metódusok

Budapesti Műszaki és Gazdaságtudományi Egyetem Közlekedés- és Járműirányítási Tanszék

```
public int getEletkor()
{
    return eletkor;
}

public void oregszik()
{
    eletkor++;
}

public void kiir()
{
    Console.WriteLine("{0} {1}", nev, getEletkor());
}
```

Konstruktorok

- Osztályok esetén ez a kezdőérték-adás nem biztos, hogy olyan egyszerű, mint volt elemi típusok esetén, ezért ebben az esetben egy függvény kapja meg az osztály inicializálásával járó feladatot.
- Ez a függvény az osztály „születésének” pillanatában automatikusan végrehajtódik, és konstruktornak vagy konstruktor függvénynek nevezzük.
- A konstruktor neve mindig az osztály nevével azonos.
- Ha ilyet nem definiálunk, a keretrendszer egy paraméter nélküli automatikus konstruktort definiál az osztály számára.
- Az osztály referencia típusú változó, egy osztálypéldány létrehozásához kötelező a **new** operátort használni, ami egyúttal a konstruktor függvény meghívását végzi el.

Konstruktorok Példa

```
public Ember(int peletkor)
{
    életkor = peletkor;
}
```

```
public Ember(Ember e)
{
    életkor = e.életkor;
    nev = e.nev;
}
```


DateTime

Típus	Tulajdonság	Magyarázat
konstruktor	<code>DateTime(Int32, Int32, Int32)</code>	Év/hónap/nap
konstruktor	<code>DateTime(Int32, Int32, Int32, Int32, Int32, Int32)</code>	Év/hónap/nap/óra/perc
static DateTime	<code>Now</code>	Pontos idő
int	<code>Year, Month, Day, Hour, Minute, Second, Millisecond</code>	Idő komponens lekérése
long	<code>Ticks</code>	0001.01.01 0:00:00 óta eltelt idő 100nanosec-ben megadva (1 ms =10 000 Tick) (1 s =10 000 000 Tick)
DateTime	<code>AddYear, AddMonth, AddDay, AddHour, AddMinute, AddSecond, AddMillisecond</code>	

String

Típus	Tulajdonság	Magyarázat
string	ToLower()	Kisbetűssé alakítás
string	ToUpper()	Nagybetűssé alakítás
string[]	Split(char[] separator)	separator mentén részekre bontás
string	Substring(Int32) Substring(Int32, Int32)	Szövegrész kinyerése
char[]	ToCharArray()	karaktertömbbé alakítás
string	Replace(Char, Char) Replace(String,String)	szövegben adott rész lecserélése
int	IndexOf(String) IndexOf(String,Int32)	Elem keresése a szövegben
static int	Compare(String strA, String strB) Compare(String strA, String strB, Boolean)	Két string összehasonlítása boolean ignorecase eredmény: <0 strA megelőzi strB-t =0 strA=strB >0 strB megelőzi strA-t

Char

Típus	Tulajdonság	Magyarázat
static char	ToLower()	Kisbetűvé alakítás
static char	ToUpper()	Nagybetűvé alakítás
static boolean	IsWhiteSpace(Char)	„Üres” mező
static boolean	IsLower(Char)	kisbetű
static boolean	IsUpper(Char)	nagybetű
static boolean	IsSymbol(Char)	szimbólum
static boolean	IsNumber(Char)	szám
static boolean	IsLetterOrDigit(Char)	szám vagy betű
static boolean	IsLetter(Char)	betű
static boolean	IsDigit(Char)	szám