OOP alapok

amit tudni kell

Sipos Miklós – Programozás I. – 2016/17/1
sipos.miklos@nik.uni-obuda.hu

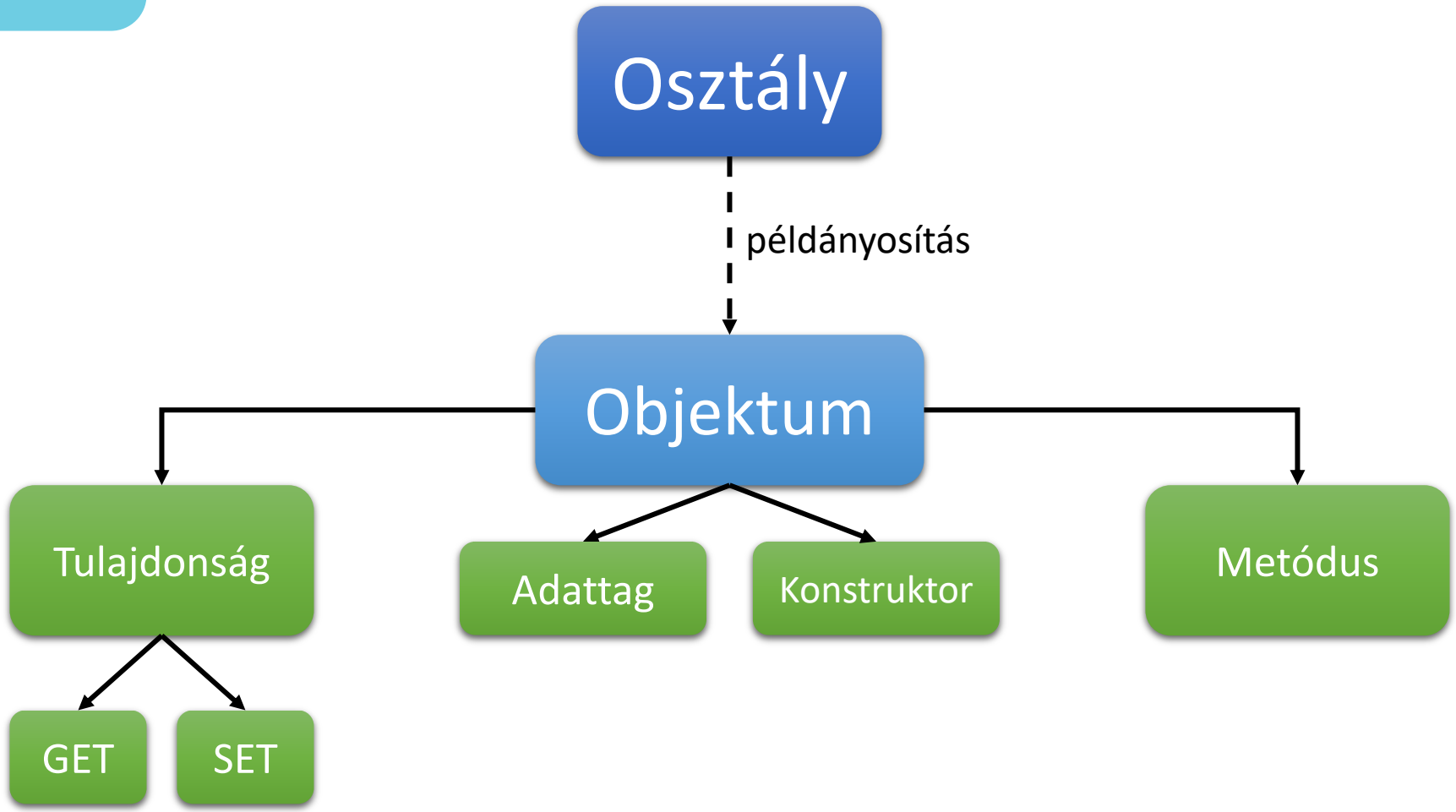
Figyelem!

A dia tartalma az órán elhangzottak **nélkül** esetenként félrevezethető lehet!

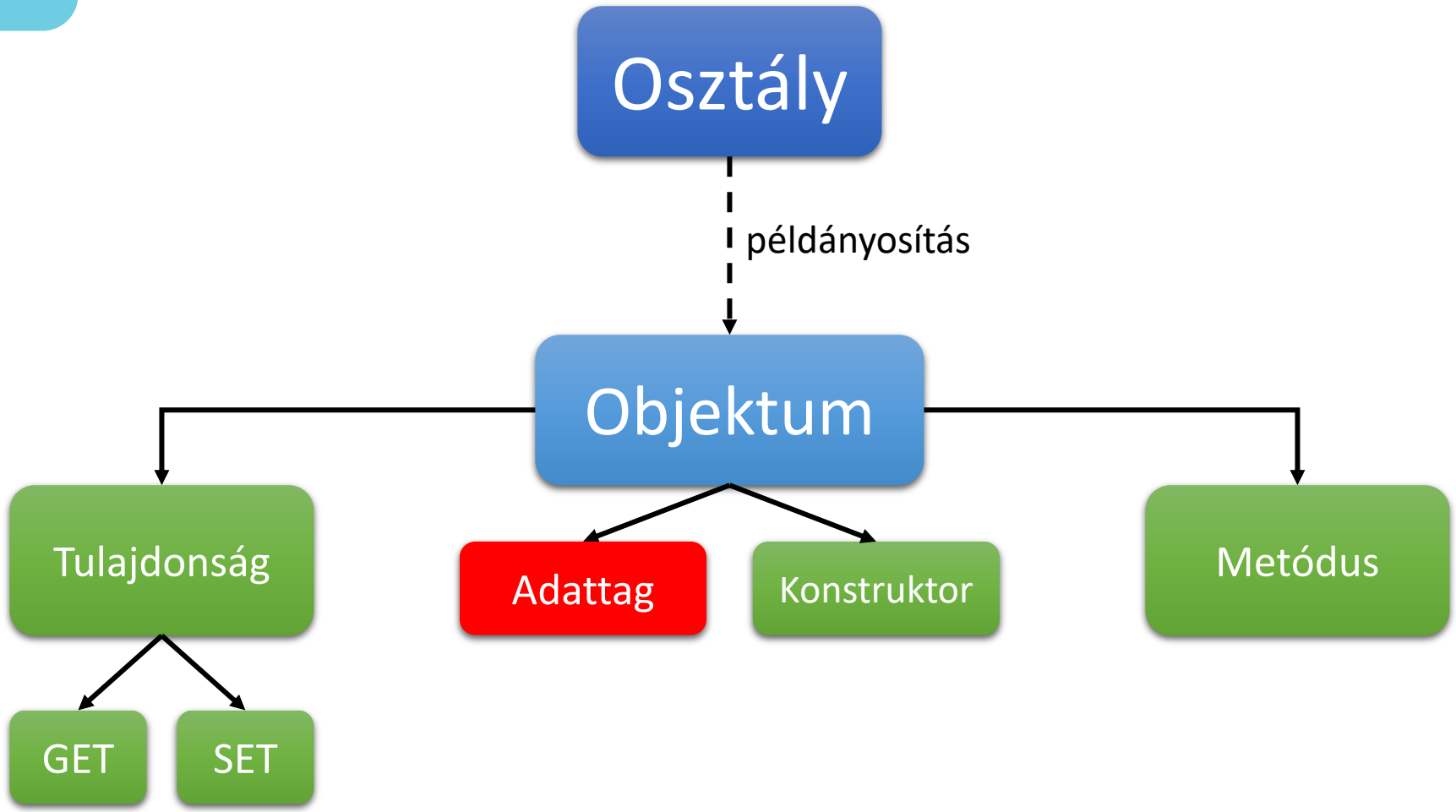
Célja, hogy az órán elmondottakat illusztrálja.

Sipos Miklós

Felépítés


Osztály
példányosítás
Objektum
Tulajdonság
GET
SET
Adattag
Konstruktor
Metódus

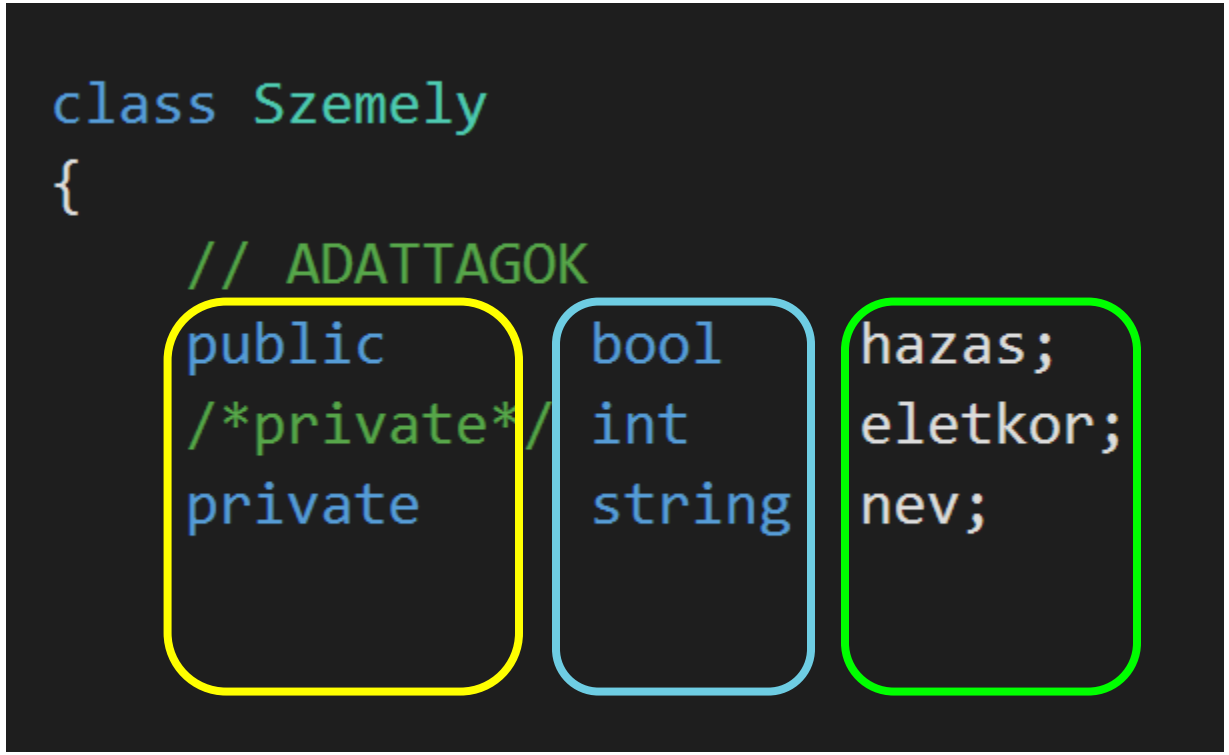
Adattag


Osztály
példányosítás
Objektum
Tulajdonság
GET
SET
Adattag
Konstruktor
Metódus

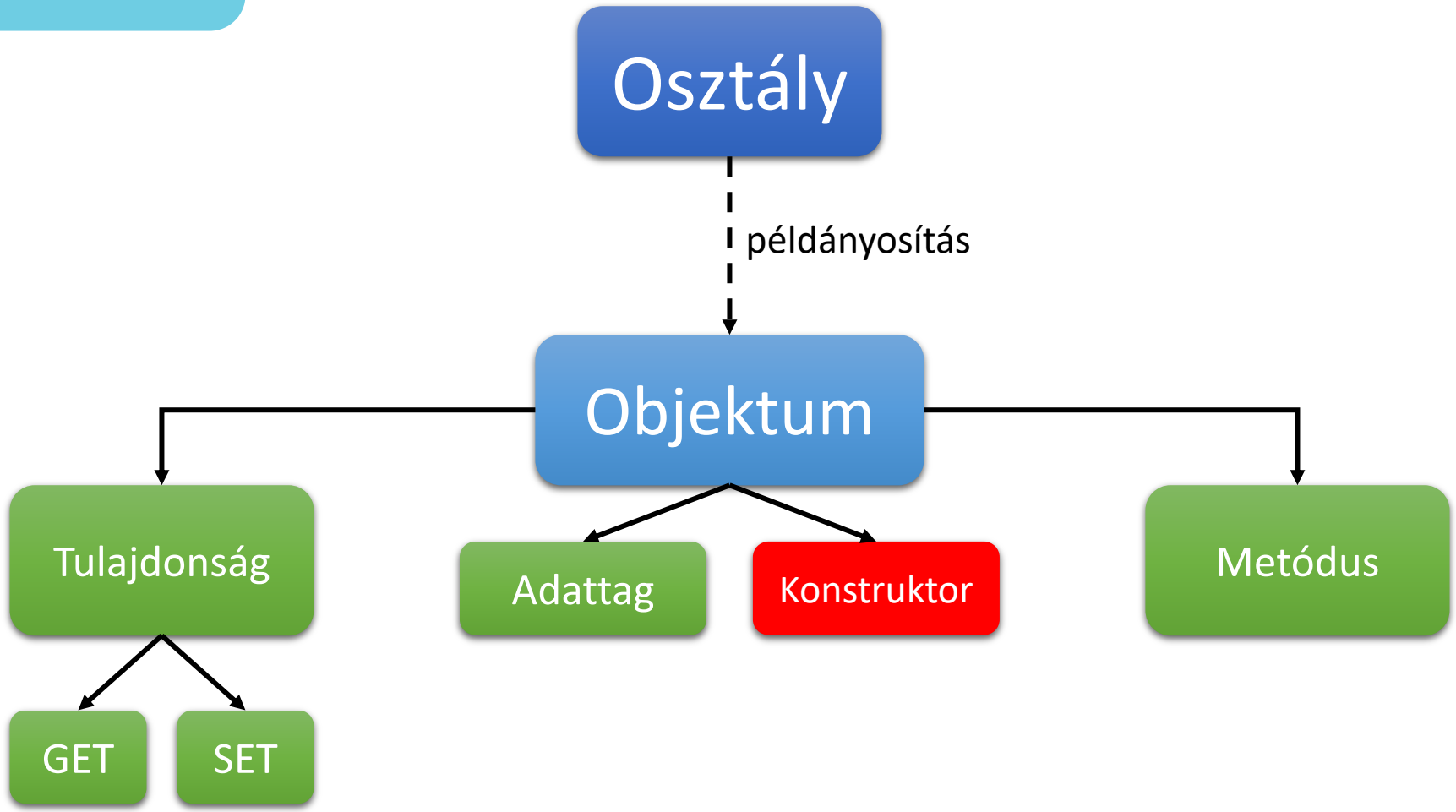
Adattag

Ha nem írunk oda semmit, az alapértelmezés private!

Célszerű nem publikusra állítani, és tulajdonság segítségével beállítani a megszorításokat.



Konstruktor



Konstruktor

Láthatóság Név

```
// KONSTRUKTOR
public Szemely(string nev, int eletkor)
{
    nev = nev;
    eletkor = eletkor;
}
```

Helytelen

```
// KONSTRUKTOR
public Szemely(string adottNev, int adottEletkor)
{
    nev = adottNev;
    eletkor = adottEletkor;
}
```

„Helytelen”

Működik, de célszerű a harmadik eljárást alkalmazni.
Lásd következő dia.

Konstruktor

Láthatóság

Név

```
// KONSTRUKTOR
public Szemely(string nev, int eletkor)
{
    this.nev = nev;
    this.eletkor = eletkor;
}
```

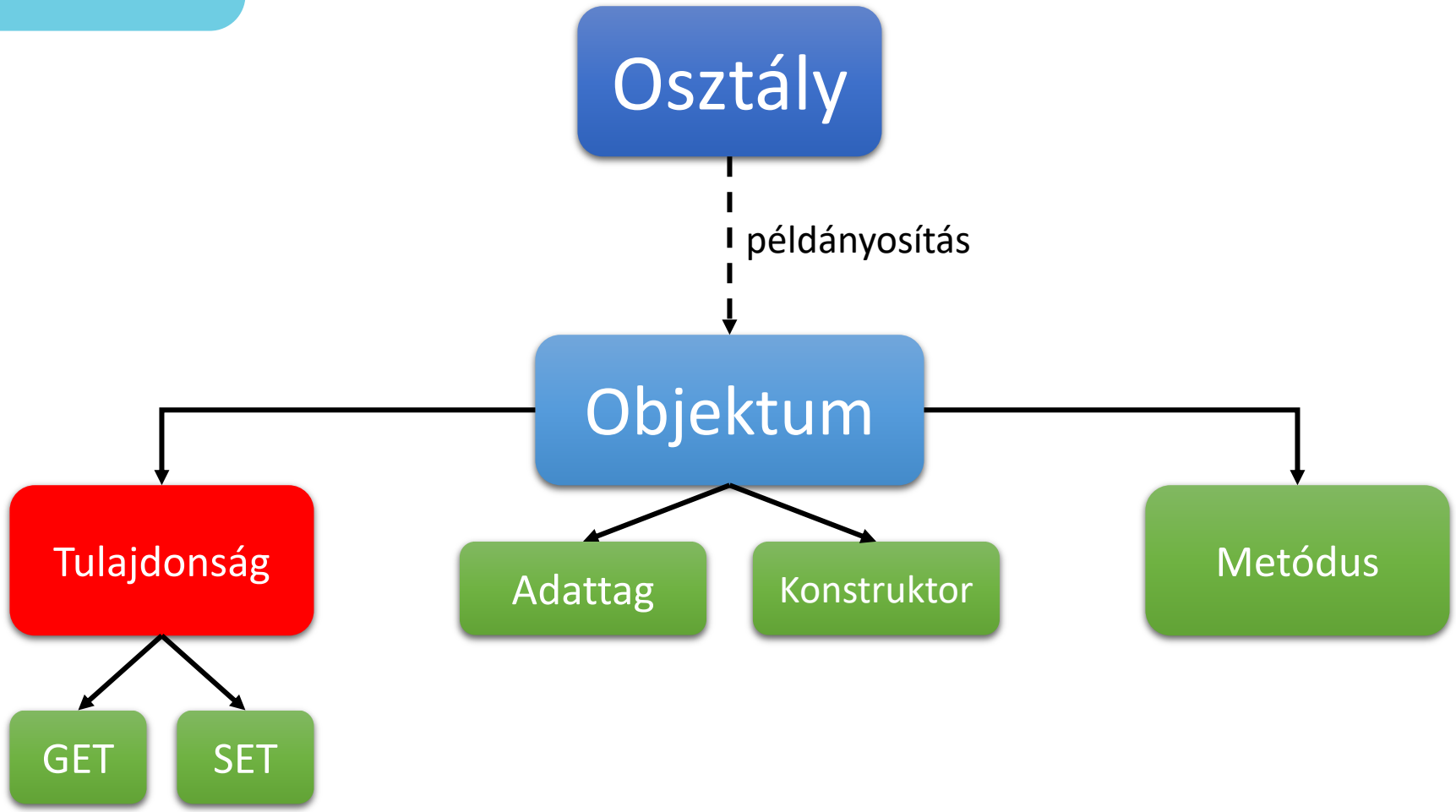
this = ez a példány

Helyes

Figyeljük meg, hogy konstruktorok esetében:

- *Nincs visszatérési érték, még void sem!*
- *A konstruktor neve, meg kell egyezzen az osztály nevével!*
- *Több konstruktor is használható, más-más bemeneti paraméterekkel. (metódus túlterhelés elven)*

Tulajdonság



Tulajdonság

Láthatóság | Típus | Név

```
// GET és SET mint metódusok
public int GetEletkor()
{
    return eletkor;
}
public void SetEletkor(int eletkor)
{
    this.eletkor = eletkor;
}
```

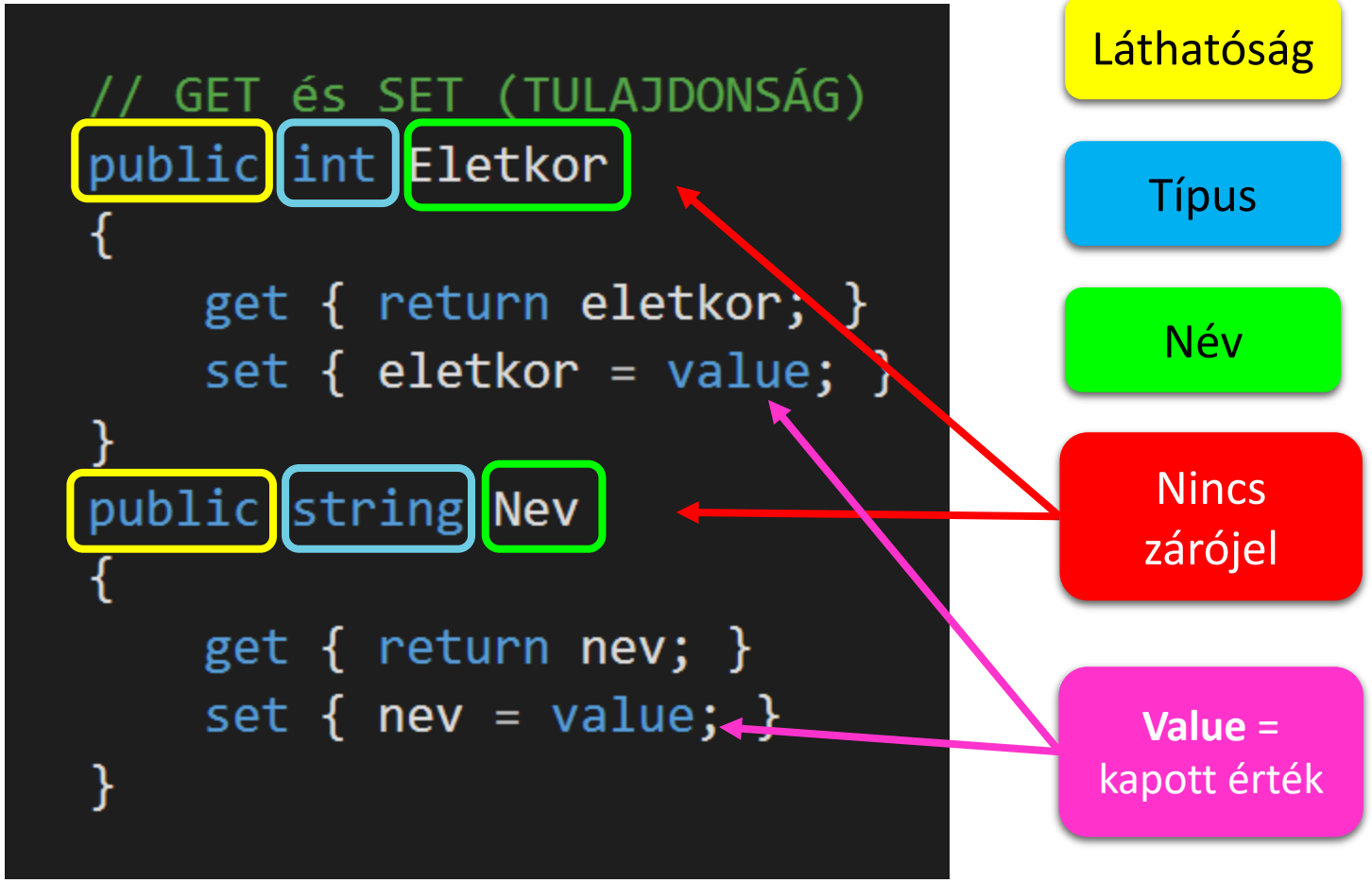
```
// GET és SET (TULAJDONSÁG)
public int Eletkor
{
    get { return eletkor; }
    set { eletkor = value; }
}
```

Nincs zárójel

Value = kapott érték

Sipos Miklós

Tulajdonság



Minden esetben publikus, a megszorítást a GET vagy SET elhagyásával tesszük meg. Lásd következő dia.

Tulajdonság

	ÍRÁS	OLVASÁS
ÍRÁS	SET	GET-SET
OLVASÁS	GET-SET	GET

Adattag, amely csak **írható** tulajdonsággal rendelkezik → SET
Adattag, amely csak **olvasható** tulajdonsággal rendelkezik → GET
Adattag, amely **írható** és **olvasható** tulajdonsággal rendelkezik → GET és SET

Tulajdonság

```
// GET és SET (TULAJDONSÁG)
public int Eletkor
{
    get { return eletkor; }
    set { eletkor = value; }
}
public string Nev
{
    get { return nev; }
    // set { nev = value; }
}
```

Írás tiltása

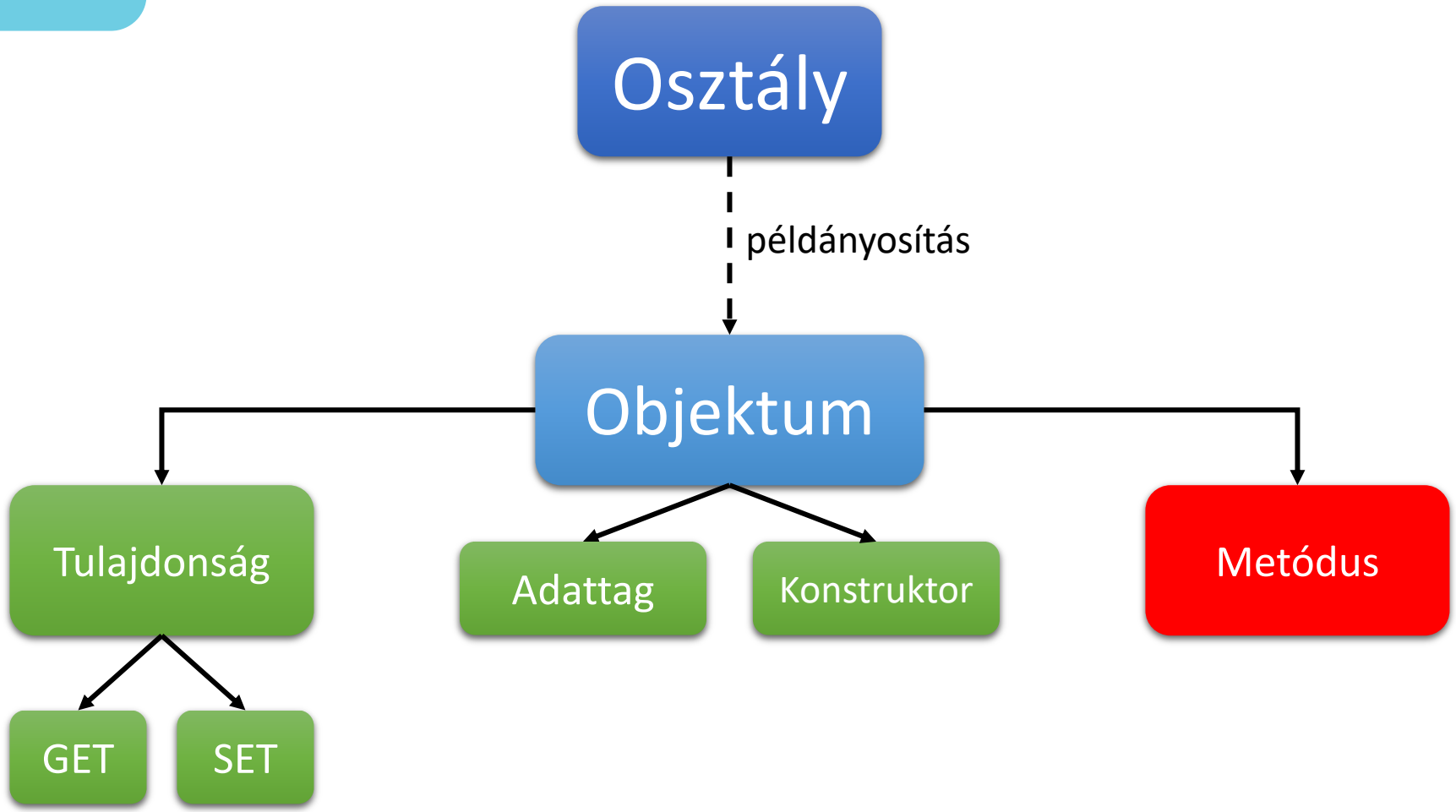
Tulajdonság

```
// GET és SET (TULAJDONSÁG)
public int Eletkor
{
    get { return eletkor; }
    set { eletkor = value; }
}
public string Nev
{
    get { return nev; }
    set { nev = "*" + value; }
}
```

Íráskor valami fix dolog végrehajtása

Túl komplikált dolgot nem érdemes idetenni, mer lassítja a tulajdonság elérését.

Metódus


Osztály
példányosítás
Objektum
Tulajdonság
GET
SET
Adattag
Konstruktor
Metódus

Metódus

Láthatóság

Típus

Név

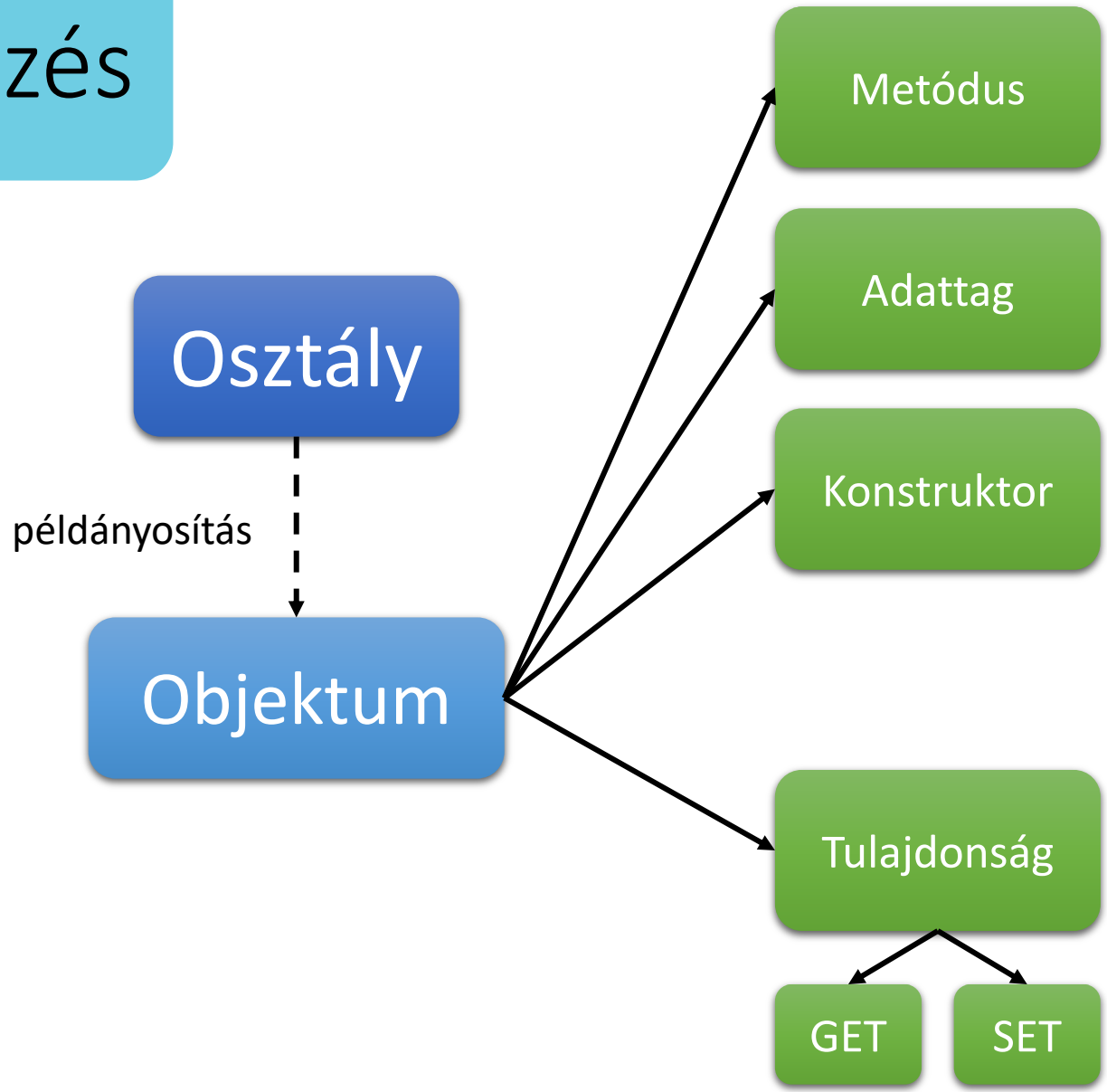
```
// További metódusok...
public void Szuletesnap()
{
    eletkor++;
    ElmultHarminc();
}
private void ElmultHarminc()
{
    if (eletkor > 30)
    {
        Console.WriteLine("Elmúlt már harminc éves.");
    }
    // else: nem csinálunk semmit.
}
```

Paraméterek ide jönnének

Public láthatósággal példány szinten is elérjük a metódusokat.

Private láthatósággal viszont csak osztály szinten, tehát osztályon belül érjük el. Ennek is meg van a maga hasznossága!

Összegzés



Az objektumhoz tartozó funkciókat tudjuk létrehozni metódusokkal.

Az objektumhoz tartozó adatokat tudjuk eltárolni változókban, amelyeket osztályok esetén adattagoknak nevezünk.

Az objektumhoz tartozó, a létrehozáskor (new kulcsszó) megadható kezdőértékeket tudjuk segítségével beállítani. Továbbá minden olyan dolog, amely a létrehozáshoz köthető. Ha nem hozunk létre konstruktort, a fordító automatikusan létrehoz egyet (e nélkül nem működne) paraméterek nélkül.
A konstruktor egy speciális metódus.

Az adattagokat tipikusan direktbe nem tesszük elérhetővé, ezért az elérést és módosítást tulajdonságokon keresztül biztosítjuk. GET esetén az adattag kiolvasását biztosítjuk, míg SET esetén az adattag módosítását.
A getter és a setter speciális metódusok. Másképp fogalmazva, a tulajdonság egy hozzáférést szabályzó réteg az adattag előtt.

Köszönöm a figyelmet!

Kérdés vagy észrevétel esetén kérem keressenek a
sipos.miklos@nik.uni-obuda.hu
elérhetőségen.