# *Programozás III*

**GENERIKUSOK, ENUM**

## TÍPUS PARAMÉTER

```
private List<Paros> adatok = new ArrayList<>();
```

Mi ez?

## JAVA – GENERIKUSOK

A **generikus** lehetőséget ad osztályok más típussal való paraméterezésére.
Pl.:

java.util

**Interface List<E>**

**Type Parameters:**

E - the type of elements in this list

A generikusan (általánosan) definiált List aktuális típus-paramétere az lehet, hogy a lista milyen konkrét típusú adatokat tartalmazzon – pl. List<Paros>

## JAVA – GENERIKUS PÉLDA

De sok más helyen is használhatjuk a generikust.

Pl. készíthetünk olyan saját generikus (általános) számológép típust, amely ugyanúgy teszi a dolgát: összead, szoroz, de egyszer egészekkel, máskor törtekkel, attól függően, hogy az Integer vagy a Double típussal paraméterezve konkretizáltuk-e.

## PÉLDA SAJÁT GENERIKUS OSZTÁLYRA

```
public class GenerikusOsszeg <T extends Number> {

   T egyik;
   T masik;

   GenerikusOsszeg(T egyik, T masik){
       this.egyik = egyik;
       this.masik = masik;
   }

   public double osszeg(){
       return (egyik.doubleValue() + masik.doubleValue());
   }

}
```

## PÉLDA SAJÁT GENERIKUS OSZTÁLYRA

```
public class GenerikusProba{

   public static void main(String args[]){
       int a=2;
       int b=3;
       GenerikusOsszeg<Integer> egy =
                       new GenerikusOsszeg<Integer>(a,b);
       System.out.println("Az összeg: " + egy.osszeg());

       float x = (float)2.4;
       float y = (float)3.2;

       GenerikusOsszeg<Float> ketto =
                       new GenerikusOsszeg<Float>(x,y);
       System.out.println("Az összeg: " + ketto.osszeg());
   }
}
```

## MÁSIK PÉLDA SAJÁT GENERIKUS OSZTÁLYRA

```
public static void main(String[] args) {
    Integer a = new Integer(1);
    Integer b = new Integer(2);
    Integer c = nagyobb(a,b);
    Double ad = new Double(2.5);
    Double bd = new Double(1.25);
    Double cd = nagyobb(ad,bd);
    System.out.println("c = " + c + " cd = " + cd);
}

private static <E extends Comparable <E>> E nagyobb(E a, E b) {
    if(a != null && b != null){
        return (a.compareTo(b) > 0)? a : b;
    }else{
        return null;
    }
}
```

```
run:
c = 2 cd = 2.5
```

## MÉG KÉT FOGALOM – 1.

**Iterátor:** java.util

### Interface Iterator<E>

**Type Parameters:**

E - the type of elements returned by this iterator

**All Known Subinterfaces:**

ListIterator<E>, XMLEventReader

**All Known Implementing Classes:**

BeanContextSupport.BCSIterator, EventReaderDelegate, Scanner

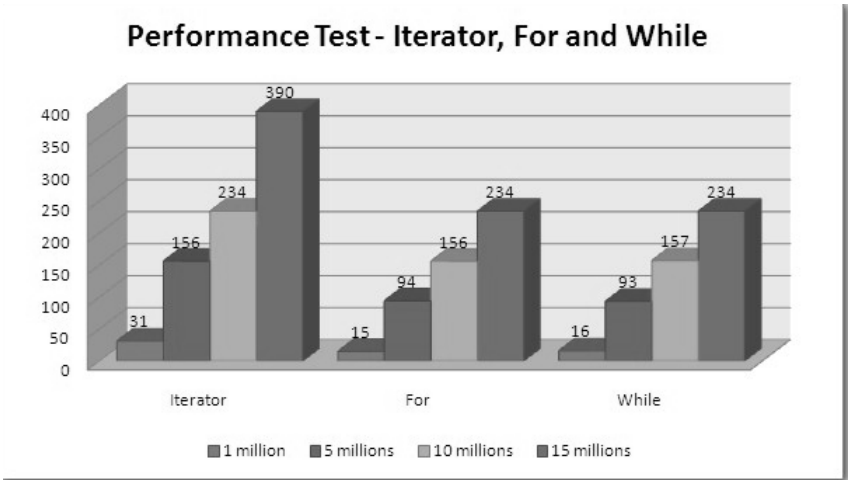
Egy iterátor segítségével végigmehetünk egy kollekción, sőt, elemet is törölhetünk belőle.

## MÉG KÉT FOGALOM – 1.

Iterátor példa

```
import java.util.List;
import java.util.ArrayList;
import java.util.Iterator;

public class IteratorPelda{

   public static void main(String args[]){
       List<String> valami = new ArrayList<String>();
       valami.add("1"); valami.add("2"); valami.add("3");
       valami.add("4"); valami.add("5");

       System.out.println("for-ral");
       for(String s : valami) {
           System.out.println(s);
       }

       System.out.println("iterator-ral");
       Iterator<String> it = valami.iterator();
       while(it.hasNext()) {
           String s = it.next();
           System.out.println(s);
       }
   }
}
```

## MÉG KÉT FOGALOM – 1. – idő teszt




http://www.mkyong.com/java/while-loop-for-loop-and-iterator-performance-test-java/

## MÉG KÉT FOGALOM – 2.

**Enumerátor:**
Az enum fix konstans értékek létrehozására használható.
Mivel típusos, így biztonságosabb mint egy int konstans.

Pl: public static final int HETFO = 1

– nem tudjuk, hogy az 1-es mit takar, és kezelni kell az érvénytelen értéket

public enum Nap { HETFO, KEDD,…

– típusos, így nem kell foglalkozni az érvénytelen értékekkel

Lehet konstruktora, és használhatunk benne final és nem final mezőket.

Enumerátor példa 1.

```
public class EnumPelda {

    public enum Nap{
        HETFO, KEDD, SZERDA, CSUTORTOK,
        PENTEK, SZOMBAT, VASARNAP
    }

    private Nap nap;

    public EnumPelda(Nap nap) {
        this.nap = nap;
    }

    public void milyenNap(){
        switch(nap){
            case KEDD: {
                System.out.println(nap + " a legjobb nap, mert Java előadás.\n");
                break;
            }
            case SZERDA:
            case CSUTORTOK :{
                System.out.println(nap + " is jó, mert Java gyakorlat.\n");
                break;
            }default: System.out.println(nap + ". Mit ér a nap Java nélkül?\n ");
        }
    }
```

## MÉG KÉT FOGALOM – 2.

```
public static void main(String[] args){
    EnumPelda kedd = new EnumPelda(Nap.KEDD);
    kedd.milyenNap();
    EnumPelda csutortok = new EnumPelda(Nap.CSUTORTOK);
    csutortok.milyenNap();
    EnumPelda pentek = new EnumPelda(Nap.PENTEK);
    pentek.milyenNap();
}
```

```
run:
KEDD a legjobb nap, mert Java előadás.

CSUTORTOK is jó, mert Java gyakorlat.

PENTEK. Mit ér a nap Java nélkül?
```

## MÉG KÉT FOGALOM – 2.

Enumerátor példa 2.

```
public class EnumPelda2 {
       public enum Nap{
        HETFO("Nehezen indul"), KEDD("Fárasztó"),
        SZERDA("Végre Java"), CSUTORTOK("Hurrá, Java"),
        PENTEK("Jön a hétvége :)"),
        SZOMBAT("Hét vége"), VASARNAP("Jaj, mindjárt hétfő");

        private String tulajdonsag;

        private Nap(String tulajdonsag) {
            this.tulajdonsag = tulajdonsag;
        }

        public String getTulajdonsag() {
            return tulajdonsag;
        }
    }
}
```

## MÉG KÉT FOGALOM – 2.

```
class indito{

    public static void main(String[] args){
        Nap[] napok = Nap.values();
        System.out.println("A hét napjai: ");
        for(Nap nap: napok){
            System.out.println(nap + " tulajdonsága: " +
                               nap.getTulajdonsag());
        }
    }
}
```

```
run:
A hét napjai:
HETFO tulajdonsága: Nehezen indul
KEDD tulajdonsága: Fárasztó
SZERDA tulajdonsága: Végre Java
CSUTORTOK tulajdonsága: Hurrá, Java
PENTEK tulajdonsága: Jön a hétvége :)
SZOMBAT tulajdonsága: Hét vége
VASARNAP tulajdonsága: Jaj, mindjárt hétfő
```

## JAVASLATOK AZ ENUM-HOZ

http://javarevisited.blogspot.hu/2011/08/enum-in-java-example-tutorial.html

http://blog.pengyifan.com/how-to-extend-enum-in-java/