Java VIII. – Parametrikus polimorfizmus

Probléma: nem túl kényelmes számunkra az, hogy a tömbök fix hosszúságúak

String[] names = { "Tim","Jerry" }; names[0] = "Tom";

String mouse = names[1];

String[] trio = new String[3];

trio[0] = names[0];

trio[1] = names[1];

trio[2] = "Spike";

names = trio;

Megoldás: java.util.ArrayList, lényegében a vector Java megfelelője

ArrayList<String> names = new ArrayList<>();

names.add("Tim");

names.add("Jerry");

names.set(0, "Tom");

String mouse = names.get(1);

names.add("Spike");

A szintaxis egy kicsit bonyolultabb, de egy kényelmesebb könnyebben kezelhető felületet biztosít az ArrayList a programozónak, miközben megőrzi a tömbök hatékonyságát (mivel a belső megvalósítás ugyan az a tömbös megvalósítás, ami az első kódrészletben volt).

Itt az ArrayList<String>-et (ejtsd: *ArrayList of String*) paraméterezett típusnak nevezzük.

Példa egy generikus osztályra:

package java.util;

public class ArrayList<T> {

public ArrayList(){ ... }

public T get( int index ){ ... }

public void set( int index, T item ){ ... }

public void add( T item ){ ... }

public T remove( int index ){ ... }

...

}

Metódusok is lehetnek generikusak:

import java.util.\*;

class Main {

public static <T> void reverse( T[] array ){

int lo = 0, hi = array.length-1;

while( lo < hi ){

T tmp = array[hi];

array[hi] = array[lo];

array[lo] = tmp;

++lo; --hi;

}

}

public static void main( String[] args ){

reverse(args);

System.out.println( Arrays.toString(args) );

}

}

A generikusok használatával nagy mértékben csökkenthetjük a kódredundanciát, ezáltal jobb minőségű kódot írhatunk.

Fontos megjegyezni viszont, hogy Java-ban csak referenciatípusokat használhatunk a generikus típusok felparaméterezésére. A nyelv azonban biztosít olyan csomagoló osztályokat (*wrapper class*), mint az Integer osztály mely segítségével primitív értékeket referencia típusokban tudunk tárolni.

ArrayList<Integer> numbers = new ArrayList<>();

numbers.add( Integer.valueOf(7) );

Kényelmi okokból automatikus ki és be csomagolás történik.

ArrayList<Integer> numbers = new ArrayList<>();

numbers.add(42); // auto-boxing: int -> Integer

int fourtytwo = numbers.get(1); // auto-unboxing: Integer -> int

További generikus adatszerkezetek:

* láncolt lista: LinkedList<>
* halmaz: HashSet<>
* leképezés: HashMap<>