Programare avansata pe obiecte – laborator 2

Diana Maftei

diana.maftei@endava.com https://github.com/DianaMaftei/pao_lab_2022

O clasa poate contine:

- Field-uri: tin starea
- Metode: manipuleaza starea si efectueaza operatii
- Constructori: crearea obiectelor, initializare
- Ca sa mostenim o clasa folosim cuvantul cheie extends
 - 1. Toate clasele din java sunt copii ai clasei Object (java.lang package)
 - toString
 - hashCode
 - equals
 - 2. Java nu suporta mostenirea multipla

Objecte

- Cand declaram un obiect, alocam spatiu pentru a stoca referinta catre acesta
- Crearea unui obiect se face prin constructor
 - 1. prin cuvantul cheie new, cream obiectul si ii asignam o referinta
 - 2. by default, compilatorul creaza un constructor fara parametri
 - 3. cand invocam in clasa copil un constructor al clasei parinte folosind cuvantul cheie super, acest apel trebuie sa fie prima linie din constructorul clasei copil
- This se refera la starea curenta a obiectului
- Null nu pointeaza catre nimic (niciun bloc de adresa) dar poate fi asignat referintelor
- Cuvantul cheie **super** este folosit pentru a accesa variabile si metode din clasa parinte

Metode

- Cand nu returnam nicio valoare, folosim return type void
- Lista de parametri e optionala
- Putem define o metoda cu acelasi nume in aceeasi clasa, dar ea trebuie sa difere prin numarul si/sau tipul parametrilor -> acest proces se numeste **supraincarcare**
- Putem define o metoda cu aceeasi semnatura ca cea din clasa parinte in clasa copil -> acest proces se numeste suprascriere

Equals and Hashcode

- Folosim **equals** pentru a verifica daca doua obiecte sunt egale. Aceasta egalitate se poate face:
 - 1. Shallow: doar verificam daca e vorba de aceeasi referinta
 - 2. Deep: comparam si starea obiectelor pe baza membrilor sai

- Folosim **hashcode** mai ales in colectii (HashMap, HashSet, HashTable). Aceasta metoda trebuie implementata in fiecare clasa unde implementam si equals
- Daca 2 obiecte sunt egale, inseamna ca ele au si acelasi hashcode; viceversa nu este valabila!

Modificatori de acces pentru datele membre/metode

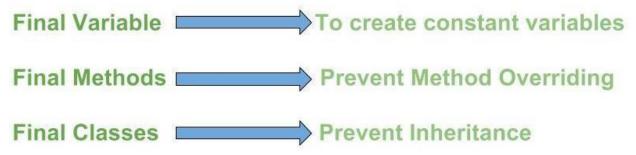
Specificator	Clasa	Subclasa	Pachet	Oriunde
Private	Х			
Protected	Х	Х	Х	
Public	Х	Х	Х	Х
Default	Х		Х	

- Este o buna practica sa accesam field-urile unei clase prin metode de tip getter si setter

Modificatorii static si final pentru datele membre/metode

Final

- Putem folosi pe clase, variabile si metode



Static

- Putem folosi pe clase, variabile, metode dar si blocuri
- Nu e nevoie de o instanta o clasei ca sa le putem accesa
- Static aplicat pe o variabila inseamna ca valoarea acesteia este impartita intre toate instantele clasei; orice modificare se va reflecta peste tot
- Metodele statice pot chema doar alte metode statice si pot contine doar date statice
- Spre deosebire de C++ in Java intalnim si blocuri statice dar nu putem avea variabile locale statice!
- Blocurile statice sunt de obicei folosite pentru initializarea variabilelor si se executa o singura data, cand se initializeaza clasa

E o buna practica ca variabilele de tip constante sa fie si final si static! Ca si naming convention, folosim litera mare si daca sunt mai multe cuvinte le separam cu _.

Ordinea initializarii

- 1. Blocurile statice, in ordinea aparitiei lor
- 2. Fieldurile si blocurile de initializare; ele se executa inaintea constructorilor dar nu inainte ca acesta sa fie chemat
- 3. Constructori

Exercitii

- 1. Write a program to create a Person object, with the following attributes: name as string, surname as string, age as int, identity number as long, type as string. Define a constructor for this class as well as accessors and mutators for all the attributes. Create two objects of type Person and display the information for them on separate lines.
- 2. Write a program to create a Room object, the attributes of this object are room number, room type and room floor. Define a constructor for this class as well as accessors and mutators for all the attributes. Create two objects of type and display the information for them on separate lines.
- 3. Write a program to create an object Subject with the following attributes: room as Room, noOfStudents as integer, teacher as Person. Define a constructor for this class as well as accessors and mutators for all the attributes. Create two objects of type Subject and display the information for them on separate lines.
- 4. Implement a Singleton class as you learned in the course.