

Notes MAP - 2023

Anchui

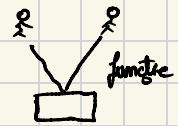
Lormin



Introducere în JAVA

Concpte OOP

- abstracțarea = "ascundem" anumite funcționalități pe care nu e nevoie să le stiu
- incapsularea = timem datele în funcții separate de exterior
- moștenirea = dacă vore unei clase să preia prop din alte clase
- polymorfism = putem crea mai multe copii ale acelorși metode
- clasa = un tip de date
- obiect = o instanță a unei clase
- mesaj = felicit de obiect pt. a comunica



ex: evitare suprascriere în VB

autostatism

magazin tractor

ex: adăugăm la magazinul VIN

Notiuni introductive

- `System.out.println();` → print on screen
 - ↳ System command
 - ↳ print line
 - ↳ output
- declarare variabile
 - la fel ca în C++
 - `String driver; driver = "Jou";`

- String things
 - `sin.length();` : dimensiune string
 - `sin = sin.toUpper();` \Rightarrow Jon
 - `sin m = sin1 + sin2;` Jon, Popescu \Rightarrow JonPopescu
 \hookrightarrow se poate face și `+ ' ' +`
- Tipuri de date
 - ca cele din C++
 - void NU este tip de date
 - byte : 1 octet (8 biti)
- Operatori
 - aritmetici: `*`, `/`, `%`, `+`, `-`
 - relaționali: `<`, `>=`, `<=`, `>`, `==`, `!=`
 - logici: `!!`, `&&`, `^`, `||`, `!>`
 \hookrightarrow OR logic \hookrightarrow AND logic
 - ternar: `? : (m > 2 == 0) ? "Per" : "Impar"`
 - accesare, apelare: `., [], ()`
 - inc/dec: `expr ++`, `expr --`
 - Shift: `<<`, `>>`, `>>>`
 \hookrightarrow ca `>>`, dar pe numere fără semn

! `double fumat = e * n;`
`int approx = (int) fumat;`
- variabile și constante
 - final int max = 100;
 \hookrightarrow constantă
 - var list = new Arraylist<String>();
 \hookrightarrow variabilă
- tablouri multidimensionale
 - tip `[]` nume_tablou; // declarare
 - `nume_tablou = new tip [dim];`

- vect.length; // dim. n_o
- tablouri N-dimensionale
 - int t[3][3] a;

Clase în Java

[public] [abstract] [final] class NumarClasa {

...

}

! semn-um fișier .java, deoarece poate fi publică

! fără ; la sf. clasei

→ public: membru accesibil tuturor
 → private: membru accesibil doar clasei

→ protected: membru accesibil doar clasei și subclaselor

- this. nume = nume; // this - referind la obiectul curent

→ moștenirea claselor

class Subclasa extends SuperClasa {

...

}

Interfețe

- sunt implementate de clase
- conțin doar metode abstracte, default și statice.

Genericitate

- cast - convertirea valorilor dintr-un tip de date în altul

ex: int value = 10;
double dValue = int value;
e ok pt că int < double

double dValue = 10.5;
int intValue = (int) dValue;

- tipuri genice - ca în C++ (public class Node<T>..)
 - ↳ E - element, K - key, N - number, T - type, V - value

Singleton → permite restriționarea numărului de instantieri ale unei clase la un singur obiect

- ex: pt. obiect tip Factory.

• 2 abordări → Eager Object instantiation

Eager object instantiation

obiectul este încremat în memorie în eșant în care poate va fi folosit

obiectul este încremat în memorie doar este folosit.

if (instance == null)

Exceptii

try h

{ . . .
} catch (TipExcepție e)

System.out.println(e);

}

- throw TipExcepție

- return face proprie clase de exceptie

-ex: public class ExceptieProprie extends RuntimeException {

{ . . .
}

Stream → orice cursă sau consumator de date care este capabil să producă sau să primească unități de date

abstract int	read()	Reads the next byte of data from the input stream.
int	read(byte[] b)	Reads some number of bytes from the input stream and stores them into the buffer array b.
int	read(byte[] b, int off, int len)	Reads up to len bytes of data from the input stream into an array of bytes.
void	reset()	Repositions this stream to the position at the time the mark method was last called on this input stream.
long	skip(long n)	Skips over and discards n bytes of data from this input stream.

void	close()	Closes this output stream and releases any system resources associated with this stream.
void	flush()	Flushes this output stream and forces any buffered output bytes to be written out.
void	write(byte[] b)	Writes b.length bytes from the specified byte array to this output stream.
void	write(byte[] b, int off, int len)	Writes len bytes from the specified byte array starting at offset off to this output stream.
abstract void	write(int b)	Writes the specified byte to this output stream.

- FileInputStream / FileOutputStream

JAVA 8

Interfețe funcționale

- conține doar o metodă abstractă

@ FunctionalInterface

interface Operatie {

 int adunare (int a, int b);

}

- permite crearea de instanțe folosind expresii lambda
- poate conține și metode default / statice

Metode statice

- nu depind de o instanță a unei clase
- sunt denumite direct prin numele interfeței
- sunt apelate folosind numele interfeței

-ex: interface Utilitati {

 static void afiseaza (String mesaj) {
 System.out.println(mesaj)
 }

Utilitati.afiseaza ("Mesaj");

Metode predefinite

- implementare implicită pt interfata
- pot fi suprascriere de clasele ce implementează interfața

ex: interface dist<T> {

 default void oflo();

 System.out.println("mesaj" + size());

 }

class lista implements dist<String> {

 @Override

 public int size() {

 return 5;

 }

}

claim:

Kirita lista = new lista();

lista.oflo();

Functii lambda

↳ funcție anonimă

- nu este legată de un identificator

ex: (int a, int b) → a+b

- simplificare expresii / înlocuire bloacei de cod

- folosite în operațiile pe Stream (filtrare, mapare, reducere)

ex: `list <String> lista = Array.asList("A", "B");`
`lista.forEach(System.out::println);`

Metode referință

Referință Obiect :: nume metoda

Interfețe funcționale din Java

- Predicate \rightarrow o condiție sau un test asupra unui obiect
 \rightarrow `test(T t)`
- Consumer \rightarrow o acțiune sau o operatie asupra unui obiect
- Function \rightarrow primește date de intrare și returnază un rezultat.
 \rightarrow `R apply(T t);`
- Supplier \rightarrow furnizor de valori, nu primește argumente
 \rightarrow `T get();`
- Comparator \rightarrow utilizat în sortarea elementelor
 \rightarrow returnă un rezultat negativ, zero sau pozitiv
 \rightarrow `int compare(T o1, T o2);`
- Optional \rightarrow gestionare situațiilor în care o val. poate lipsi
sau poate fi prezentă
 \rightarrow evite valoare null

Stream \rightarrow permite operațiuni de procesare a datelor
într-un mod elegant și funcțional

\rightarrow Operațiuni: filter, map \rightarrow transf. elem în alte valori,
distinct \rightarrow elem. dupăcare, sorted,
limit \rightarrow limitare nr. elem., skip

→ operatiuni terminale: `forEach`, `toArray`, `collect`, `reduce`,
min, max, count, findAny, match

Java FX

- clase și interfețe ce oferă suport de creare aplicații Java ce se pot proiecta/implementa pe diverse platforme
- Interfețe grafice + CSS

Java FX

- window = stage
 - ↳ content = scene
- stage • — setTitle

StackPane - layout

layout - getChildren()

• add()

stage → layout → scene