TEMA:Constructori parametrizati,modificatori de acces. Incapsularea.Pachete.Mostenirea

18.05.2021

CONSTRUCTORI CU PARAMETRI:

//Constructori supraincarcati  
**public Circle(){** radius=1.0;  
 color="red";  
}  
  
**public Circle(double r){**  
 radius=r;  
 color="red";  
}  
  
**public Circle(double r,String c){**  
 radius=r;  
 color=c;  
}

public class TestCirlce {  
 public static void main(String[] args) {  
  
 //Cream obiect al clasei circle cu numele c1  
 Circle c1=new Circle(2,"blue");  
 System.*out*.println("Radius="+c1.getRadius()+" Culoare="  
 +c1.getColor()+" Suprafata="+c1.getArea());  
  
 Circle c2=new Circle(3);  
 System.*out*.println("Radius="+c2.getRadius()+" Culoare="  
 +c2.getColor()+" Suprafata="+c2.getArea());  
  
 Circle c3=new Circle();  
 System.*out*.println("Radius="+c3.getRadius()+" Culoare="  
 +c3.getColor()+" Suprafata="+c3.getArea());  
  
 }  
}

MODIFICATORI:

* **Public**-deschis pt uz extern.
* **Private-**inchis,campul e disponibil numai in clasa in care e declarata.(Pt a folosi variabila private folosim setter si getteri)
* **Protected**-protejat,campul nu e disponibil pt uz extern,e disponibil pt clasa si mostenitoriil si clasele pachetului current.
* **Default:** The class is only accessible by classes in the same package. This is used when you don't specify a modifier. You will learn more about packages in the Packages chapter

INCAPSULAREA:

The meaning of **Encapsulation**, is to make sure that "sensitive" data is hidden from users. To achieve this, you must:

* declare class variables/attributes as private
* provide public **get** and **set** methods to access and update the value of a private variable

GETTERI SI SETTERI SINTAXA:

* Prin getteri si setter avem acces la variabilele private in afara clasei.
* (get si set sunt mereu **metode** **publice**)

public double getSalariu(){  
 return salariu;  
}  
public void setSalariu(double s){  
 this.salariu=s;  
}

PACHETE:

Un pachet în Java este folosit pentru a grupa clase legate.

Gândiți-vă la acesta ca la un folder dintr-un director de fișiere.

Folosim pachete pentru a evita conflictele de nume și pentru a scrie un cod mai ușor de întreținut.

Pachetele sunt împărțite în două categorii:

* Pachete încorporate (pachete din API-ul Java)
* Pachete definite de utilizator (creați propriile pachete)

Sintaxa:

import *package*.*name*.*Class*; // Import a single class

import *package*.*name*.\*; // Import the whole package

MOSTENIREA:

În Java, este posibil să moștenim atribute și metode de la o clasă la alta. Grupăm „conceptul de moștenire” în două categorii:

* subclasă (copil) - clasa care moștenește de la o altă clasă
* superclasă (părinte) - clasa din care este moștenită

Pentru a moșteni dintr-o clasă, utilizați cuvântul cheie „**extends”**

Example

class Vehicle {

protected String brand = "Ford"; // Vehicle attribute

public void honk() { // Vehicle method

System.out.println("Tuut, tuut!");

}

}

class Car extends Vehicle {

private String modelName = "Mustang"; // Car attribute

public static void main(String[] args) {

// Create a myCar object

Car myCar = new Car();

// Call the honk() method (from the Vehicle class) on the myCar object

myCar.honk();

// Display the value of the brand attribute (from the Vehicle class) and the value of the modelName from the Car class

System.out.println(myCar.brand + " " + myCar.modelName);

}

}