Objektno-orijentisano programiranje

Uvod u Javu

Nevena Ćirić
www.matf.bg.ac.rs/~nevena_ciric

OOP kao nov način pisanja programa

- Programska paradigma (način programiranja, programski stil) koji ste radili na prethodnim kursevima kroz programske jezike C i Asembler je imperativna/proceduralna paradigma.
- Def: Osnovni pojam imperativnih programskih jezika je NAREDBA čijim se izvršavanjem menja stanje programa. Naredbe su grupisane u procedure i funkcije.
- Def: Funkcija je programska celina koja ulazne podatke (argumente) transformiše u odgovarajuće izlazne podatke (povratna vrednost).
- Def: Procedura je programska celina koja izvršava neki proces (niz naredbi) ali ne daje nikakav izlaz kao rezultat svog izvršavanja.

OOP kao nov način pisanja programa

- Def: Objektno-orijentisano programiranje je programska paradigma zasnovana na pojmu OBJEKATA. Izvršavanje programa se zasniva na promeni stanja pojedinačnih objekata i njihovoj međusobnoj interakciji.
- Def: Objekat je integralna celina podataka (atributa) i procedura (metoda) za rad sa njima.

METOD = PROCEDURA KOJA PRIPADA NEKOM OBJEKTU

OOP kao nov način pisanja programa

- Def: Klasa je skup objekata sa zajedničkim svojstvima, koji se ponašaju na isti način. Klasa definiše šablon za kreiranje objekata, tj. opisuje strukturu objekta.
- Primerak (instanca) klase je konkretan objekat date klase. Klasom je definisan tip objekata, a svaki objekat primerak (instanca) ima konkretne vrednosti podataka.
  - klasa u Javi~ struktura u C-u
  - objekat u Javi ~ promenljiva u C-u
- Definisanje klase u Javi odgovara definisanju novog tipa podataka u C-u.
- U Javi je svaki objekat primerak neke klase. Objekti postoje samo tokom izvršavanja programa.

Programski jezik Java

- Tehnike objektno-orijentisanog programiranja učićemo kroz programski jezik Java.
- IntelliJ IDEA – razvojno okruženje koje ćemo koristiti
- IDE – Integrated Development Environment
- Ovaj alat omogućava brz razvoj Java aplikacija tako što nudi integrisano:
  - Editovanje
  - Kompajliranje
  - Interpretiranje
  - Debagovanje
  - Online Help

Java u odnosu na C

- C
  - Ekstenzija izvornog kôda: .c
  - Prevođenje: kompilacija
  - Kompilator: gcc
  - Ekstenzija izvršnog fajla: .out (Linux) i .exe (Windows)
- Java
  - Ekstenzija izvornog kôda: .java
  - Prevođenje: kompilacija + interpretacija
  - Kompilator: javac
  - Ekstenzija izvršnog fajla: .class

Prevođenje Java programa

Kompilacija vs interpretacija

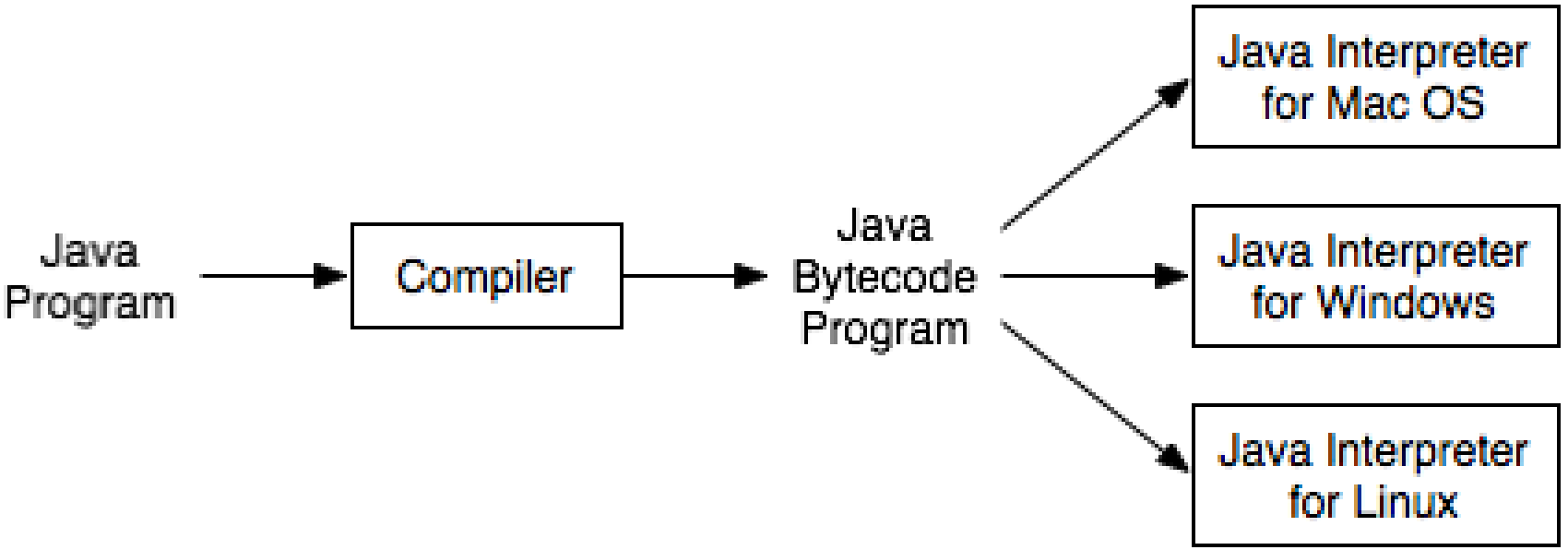
- Programi napisani na nekom jeziku visokog nivoa ne mogu se direktno izvršiti na računaru, već se moraju najpre prevesti na mašinski jezik određenog procesora.
- Alternativa kompajliranju programa sa jezika visokog nivoa u mašinski jezik radi izvršavanja programa jeste postupak koji se naziva interpretiranje. Umesto kompajlera, koji prevodi ceo program, koristi se interpretator koji istovremeno prevodi i izvršava program deo-po-deo, kako to logika programa nalaže.
- Kompajliranje može biti do mašinskog jezika ili do nekog međujezika – u Javi do Java Bajt kôda.

Java Bajt kôd i JVM

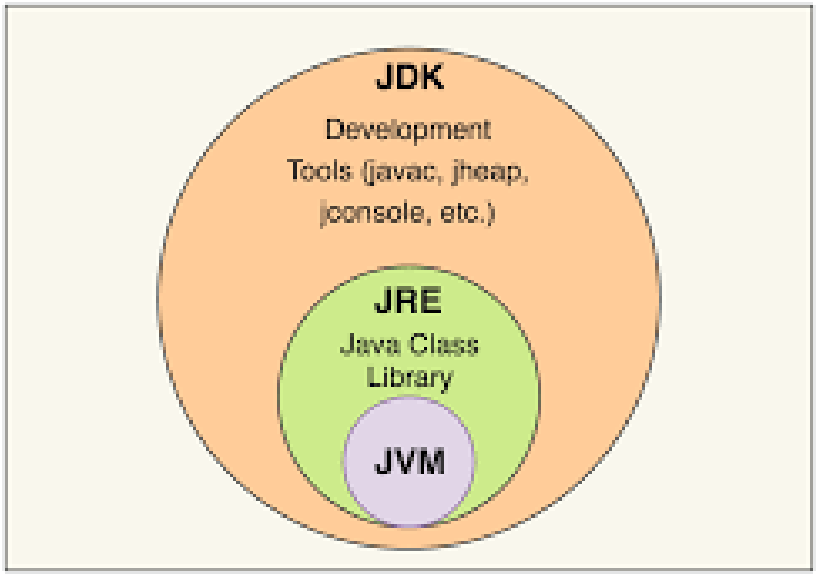
- Java Bajt kôd je međujezik apstraktne računarske mašine – Java virtuelne mašine (JVM). To je visokooptimizovan skup instrukcija koji u trenutku pokretanja programa tumači (interpretira) Javin izvršni sistem, poznat kao Java virtuelna mašina.
- Virtuelna mašina (VM) je softverska implementacija mašine (npr. računara) koja izvršava program kao "prava" mašina. Predstavlja simulaciju mašine koja se obično razlikuje od ciljne mašine (mašine na kojoj se simulacija obavlja). Postoje dve vrste:
  - Virtuelna mašina za simuliranje sistema koja obezbeđuje kompletnu sistemsku platformu koja podržava izvršavanje celog operativnog sistema (npr. VMWare, VirtualBox)
  - Virtuelna mašina za simuliranje procesa (i jezika) omogućava izvršavanje jednog programa, što znači da podržava samo jedan proces
- VIRTUELNA MAŠINA = PROGRAM KOJI OMOGUĆAVA IZVRŠAVANJE DRUGIH PROGRAMA

Prevođenje Java programa

- JVM su raspoložive za mnoge hardverske i softverske platforme.
- **Java Bajt kôd i JVM rešavaju problem prenosivosti kôda.**



JVM vs JRE vs JDK



JRE = JVM + Library Classes
JDK = JRE + Development Tools

- JRE (Java Runtime Environmet) obezbeđuje okruženje za izvršavanje Java programa (u formi Java Bajt kôda jer kompajler javac nije deo JRE već JDK). JRE koriste oni koji žele samo da izvršavaju Java programe, tj. krajnji korisnici sistema.
- JDK (Java Development Kit) je okruženje koje omogućava razvijanje Java programa (a samim tim i izvršavanje Java programa)

Java projekat

- Java projekat je skup svih fajlova koji čine jedan program i podešavanja vezanih za način pisanja, prevođenja i izvršavanja programa (npr. koja verzija Jave se koristi, podešavanja editora, količina memorije potrebna prilikom izvršavanja programa, koja klasa je 'main' klasa ili definisanje više različitih ulaznih tačaka projekta, itd.)
- Koncept projekta nije vezan za programski jezik Java. To je konstrukt koji obezbeđuje razvojno okruženje u cilju poboljšanja razvojnog procesa programa.
- Java projekti imaju strukturu direktorijuma.

Java paketi

- Java paket je grupa povezanih klasa* i pod-paketa. Koriste se da grupišu klase koje pripadaju istoj kategoriji ili obezbeđuju sličnu funkcionalnost u cilju bolje organizacije programa koji se sastoje od velikog broja klasa. Može biti:
  - ugrađeni paket - java, util, lang, io, nio, javafx, awt, swing, net, sql,..
  - korisnički definisani paket
- Svaka klasa u Javi je sadržana u nekom paketu, a ako se ne navede eksplicitno, naše klase se smeštaju u podrazumevani (default) paket koji nema ime.
- Standardne Java klase su sadržane u paketima tako što se u jednom paketu nalaze klase koje su na neki način povezane.
- Paket predstavlja jedinstveno imenovanu kolekciju klasa*. Imena klasa jednog paketa neće se mešati sa imenima klasa nekog drugog paketa zato što se imena klasa paketa kvalifikuju imenom tog paketa (npr. puno ime klase String iz paketa java.lang je java.lang.String).

* Zapravo klasnih tipova (klasa, interfejsa, enumerisanih tipova i tipova notacije), ali mi ćemo za sada govoriti samo u terminima klasa.

Java klasa

- Java izvorni kôd se uvek čuva u fajlovima sa ekstenzijom .java
- U fajlu može da bude definisano više klasa, ali najviše jedna javna (public) klasa. Ime javne klase mora da bude jednako imenu fajla. Uobičajeno je da se svaka klasa stavlja u poseban fajl.
- Klasa iz paketa može pristupati proizvoljnoj klasi iz istog paketa. Klase iz drugih paketa mogu, a ne moraju biti dostupne.
- Ako želimo da klasa paketa bude dostupna i izvan njega, potrebno je deklarisati klasu koristeći ključnu reč public.

Metod main()

- Svaka Java aplikacija sadrži klasu koja definiše metod main(). Ime te klase je argument koji prosleđujemo Java interpreteru (JVM).
- Metod main() mora imati fiksnu formu i ukoliko ona nije ispoštovana, neće biti prepoznat od strane Java interpretera (JVM) kao metod od koga kreće izvršavanje programa.
- Najjednostavnija Java aplikacija sastoji se samo od jedne klase koja ima samo jedan, main() metod.

Metod main()

- Deklaracija main() metoda je uvek oblika:

  public static void main(String[] args)

  - void – znači da funkcija ne vraća nikakvu vrednost
  - public – da je metod globalno dostupan
  - static – dostupan i kada ne postoje objekti klase
  - sa args.length možemo odrediti dužinu niza

System.out.println()

- System je ime standardne klase koja sadrži objekte za rad sa standardnim U/I uređajima (tastatura za ulaz, monitor za izlaz).
- Objekat out predstavlja standardni izlazni tok i on je statički član klase System, što znači da on postoji čak i kada ne postoji nijedan objekat tipa System.
- System.out ← notacija: out je član klase System
- println() metod pripada objektu System.out