Cvičenie k PB162 Programování v jazyku Java

Slajdy k druhému cvičeniu

Rastislav Mirek

Fakulta informatiky, Masarykova univerzita

1. októbra 2009

Dneštné cvičenie

- Opakovanie: Trieda, objekt, atributy, metody, subori .java a .class, konvencie, prikazy prikazoveho riadku
  - možnosť ziskať body do cvičení za aktivitu a správne odpovede
- Životny cyklus objektu, konštruktory, prikaz new, objektové premenne, lokalne premenné a ich platnosť, lokalne premenné verzus atributy, **if** a **for**, metóda main()
  - pýtať sa, pýtať sa, pýtať sa
- Ukážkový príklad
  - požiadať o spomalenie a dovysvetlenie ak nieco nie je jasné
- Príklad za 3 body do cvičení

Opakovanie zakladných pojmov

- Mali by ste vedieť:
  - Čo je to trieda ?
  - Čo je to objekt ?
- Trieda je zovšeobecnením objektov
- Objekty sú inštanciami triedy - jej konkretizáciami.
- Trieda má atributy(vlastnosti) a metódy(činnosti, ktoré môže vykonávať)
- Uvedte príklad - trieda jej typické atribúty a metódy. Inštancie tejto triedy(objekty) a hodnoty atributov, ktoré môžu napr. mať

Atribúty

- Vlastnosti danej triedy
- Len pre kontext našej aplikácie relevatné vlastnosti
- Všetky inštancie(objekty) tejto triedy maju tieto vlastnosti ale jednotlive ištancie môžu mat iné **hodnoty**(samotná trieda nemá hodnoty atribútov)
- Atribúty môžu byť napr. textové, číselné či ich hodnotou môže byť objekt nejakej triedy (hovoríme o type atribútu)
- atribúty deklarujeme v triedach (obyčajne) tesne pod hlavičkou triedy
- deklarácia: <viditelnosť> <typ> <názov>;
  <viditelnosť> je u atributov vždy *private*
  <typ>
  číselný - *int*
  text - *String* (je to objekt preto velke S)
  boolovska hodnota - *boolean*
  objektový typ - nazov lubovolnej triedy, hodnotou takehoto atributu budu instancie danej triedy

Metódy

- Činnosti, ktore je možné po objektoch triedy pozadovat
- Len pre kontext našej aplikácie relevatné činnosti, ktore su objekty schopne
- Všetky inštancie(objekty) triedy maju tieto cinnosti rovnake(definovane v zdrojovom kode triedy) ale vykonavat ich mozu len konkretne objekty
- Vykonat cinnost na objekte znamena zavolat jeho metodu
- deklarácia: <viditelnosť> <navratovy typ> <názov>(<typ parametru> <nazov parametru>{, dalsie parametre oddelene ciarkami...}){<telo metody>}
  <viditelnosť> definuje ci mozu byt metody volane instanciami inej triedy
  moze mat hodnoty *private*, *protected*, *public* alebo sa klucove slovo vynecha
  zatial(druhe cvicenie pouzite vzdy public)

Metody 2

<navratovy typ> typ vysledku metody ktory metoda “vrati” moze byt niektory z typov, ktore mozu mat atributy (vid. predchadzajuci slajd) alebo specialny nulovy navratovy typ *void* - ked metoda len meni stav objektu t.j. meni jeho atributy ale nema vysledok, ktory by vratila
<názov> je vystižný anglický názov cinnosti. Pozri konvencie pre pravidla o pomenuvani
<typ parametru> je ktorykolvek z typov ktore mozu mat aj atributy (predchadzajuci slajd)
<nazov parametru> je anglicky nazov parametru (vid. konvencie)

- v tele metody pouzijeme *return* <hodnota ktoru ma metoda vratit>*;* na ukoncenie metody a vratenie pozadovanej hodnoty
- metody z navratovym typom *void* koncia automaticky zlozenou zatvorkou

Organizacia suborov a balicky

- Zdrojovy kod triedy v Jave je typicky ulozeny v samostatnom subore
  - nazov tohto suboru je vzdy nazov danej triedy (CaSe sensitive) s priponou .java
- Pri kompilacii sa pre kazdy subor s priponou .java vytvori subor z rovnakym nazvom a priponou .class
- Triedy su organizovane v balickoch
- Balicek je zaroven suborom na disku. Trieda patri do balicku prave vtedy ked jej .java subor je v umiestneny v prislusnej zlozke
- Vsetky balicky programu musia byt umiestnene v balicku, ktory ma nazov obratenej sietovej adresy organizacie pre ktoru program tvorime. Dalej sa balicky organizuju podla internych pravidiel organizacie.
- Vsetky ulohy a pisomky do predmetu PB162 ktore odovzdate maju vzdy byt v balicku cz.muni.fi.pb162

Konvencie

- Java je CaSe sensitive - Case a case su dva rozdielne identifikarory
- Nazvove konvencie:
  - vsetky pismena v nazvoch balickov su male - lower case
  - nazov triedy zacina velkym pismenom, kazde nove slovo zacina velkym pismenom ostatne su male
  - nazvi atributov a metod zacinaju malym pismenom kazde dalsie slovo v nazve zacina velkym
  - rovnako aj nazvy parametrov metod a lokalnych premennych
- Viditelnost atributov je vzdy private
- Ku kazdej triede vzdy vyplnime dokumentaciu formou java doc a nezabudneme zadat @author
- Mali by sme vyplnat aj dokumentaciu k metodam (pre ucely tohto cvicenia netreba)

Kompilacia a spustenie z prikazovej riadky

- Triedu z prikazoveho riadku skompilujeme prikazom `javac cz.muni.fi.pb162.test.ClassName.java` spustenom z adresara v ktorom sa nachadza zlozka cz ked cz.muni.fi.pb162.test je nazov nasko balicku a ClassName je nazov triedy
- po skompilovani vsetkych tried mozme spustit prikazom `java cz.muni.fi.pb162.test.ClassName`

Konstruktory

- Trieda je sablonou na vytvaranie objektov
- Životny cyklus objektu:
  - objekt je vytvoreny "zo sablony" volanim konstruktoru
  - na objekte su volane metody, hodnoty jeho atributov sa menia...
  - ak na objekt neexistuju odkazi, objekt zanika
- Konstruktor je specialna "metoda" ktore vytvara novy objekt a moze nastavit inicialny stav objektu
- Konstruktor v triede deklarujeme: <viditelnost> <nazov triedy>(<typ parametra 1> <nazov parametra 1>{, dalsie parametre...})<telo konstruktoru>
  <viditelnost> pre nase ucely zatial *public*
  <nazov triedy> nazov triedy v ktorej konstruktor deklarujeme a ktorej instanciu vytvara
  <typ parametra 1> moze byt ako pri atributoch

Konstruktory 2

- novy objekt potom vytvorime pomocou new ako keby sme volali metodu (rozdiel je len v prikaze *new*)
- Priklad:

```
public class Person{
    private int age;
    private String name;
    public Person(int initialAge, String name){
        age = initialAge;
        this.name = name;
    }
}
```

- Potom je mozne vytvorit novy objekt triedy person takto:

```
Person me = new Person(23, “Rastislav Mirek”);
```