INOVÁCIA VYUČOVANIA PREDMETOV PROGRAMOVANIA NA PEDAGOGICKEJ FAKULTE TRNAVSKEJ UNIVERZITY

Roman Horváth

Stredisko pre celoživotné vzdelávanie, Pedagogická fakulta, Trnavská univerzita
Priemyselná 4, P. O. Box 9, 918 43 Trnava, SR
e-mail: rhorvath@truni.sk

Abstract. This paper briefly explains the object-oriented programming paradigm to readers who need not be deeply interested in this area. It presents also reasons for implementation of a new teaching method at Faculty of Education, University of Trnava.

Keywords: BlueJ, object-oriented programming paradigm, visualized inheritance, role of debugging in teaching process

1. Úvod

Objektovo-orientované programovanie (ďalej OOP) je jednou z novších programovacích paradigiem. Z toho dôvodu sme sa na Pedagogickej fakulte TU (PdF TU) rozhodli prejsť na vyučovanie predmetov programovanie a algoritmy 1 a 2 podľa tejto paradigmy. Programovacím jazykom je Java, programovacím prostredím je BlueJ.

2. Paradigma objektovo-orientovaného programovania

Paradigma OOP ([1], [2], [3], [4], [5], [6]) má niekoľko výrazných znakov, ktoré pri klasickom imperatívnom programovaní ([7], [8]) nie sú badateľné (neznamená to, že nie sú použiteľné, len ich implementácia vyžaduje omnoho viac úsilia zo strany programátorov). Správnou aplikáciou hlavných čŕt OOP, ktorými sú: dedičnosť, polymorfizmus, zapuzdrovanie a abstrahovanie dát, modularita… je možné dosiahnuť produkciu kvalitnejších a spoľahlivejších programov a písať prehľadnejší zdrojový kód, čo je dôležité pri údržbe softvéru.

Návrh a implementácia vymenovaných vlastností vyplynuli zo skúseností z programátorskej praxe, najčastejšie sa vyskytujúcich chýb a z nich plynúcej nespoľahlivosti programov a z postupne sa rozvíjajúcich čiastkových riešení, ktoré pomáhali týmto chybám predchádzať. Tieto riešenia boli zárodkom nového smeru – OOP.

Pri objektovo-orientovanom programovaní programátor pracuje s celkami ako trieda či šablóna, ktoré využíva pri tvorbe inštancií resp. objektov. Pri tvorbe tried môže využiť prvú vlastnosť – dedičnosť, pod ktorou rozumieme odvodzovanie nových tried z už existujúcich tried. Odvodené triedy môžu rozšíriť vlastnosti materskej triedy, prípadne zmeniť jej správanie, čo súvisí s ďalšou vlastnosťou – polymorfizmom.

Polymorfizmus, alebo „viactvarosť“, poskytuje programátorom vyšší stupeň voľnosti pri tvorbe kódu. Je viac druhov polymorfizmu – parametrický, inkluzívny, preťažovanie, pretypovanie, prekrývanie… Každý funguje mierne odlišným spôsobom. Jednotlivé druhy polymorfizmu sú využívané pri tvorbe šablón, preťažovaní operátorov a metód odvodených tried, ale polymorfizmus umožňuje aj použitie inštancií rodičovských (všeobecnejších) tried tam, kde sú očakávané ich odvodené verzie. Programátor tak má viac voľnosti pri písaní kódu, čo však môže priniesť problémy pri typovej kontrole a pri nepozornosti viesť ku vzniku

chýb, ktoré pri klasickom imperatívnom programovaní nevznikali. V každom prípade je OOP prístup oproti svojim predchodcom kvalitatívne lepší.

Zapuzdrovanie dát do tried, rieši problém globálnych dát, ktoré mohli byť pri klasickom imperatívnom prístupe kedykoľvek omylom prepísané ľubovoľnou časťou programu. To bolo problémom najmä pri tímových projektoch. Dáta skryté v triedach sú viditeľné len pre inštancie vlastnej triedy, ktorá kompletne manažuje ich obsah. Dozerá na platnosť dát, ktoré sú zapisované do dátových štruktúr a umožňuje zamedzenie prístupu na čítanie dát neoprávnenými entitami. Písanie kódu sa týmto stáva robustnejším.

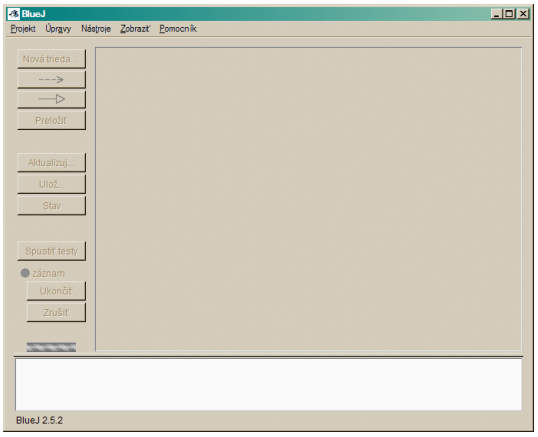
3. Inovácia vyučovania programovania na Pedagogickej fakulte TU

Výhody OOP prístupu a jeho rozšírenosť vo svete viedli k inovácii vyučovania predmetov programovania (týka sa študijných programov v kombinácii s informatikou) na PdF TU. Bolo potrebné zvoliť vhodný objektovo-orientovaný jazyk, s čím súvisel aj výber vhodného programátorského prostredia. V rámci aktuálne používaných programovacích jazykov pripadali do úvahy v podstate len tri najpoužívanejšie: C++, C# alebo Java. Rozhodli sme sa pre Javu pre možnosti jej multiplatformového použitia a pre jej rozšírenosť.

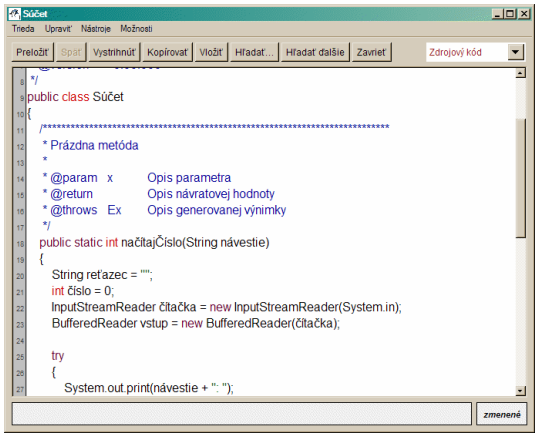
Výber vhodného programovacieho prostredia bol podmienený účelom použitia. Podarilo sa nám nájsť prostredie, ktoré bolo priamo vyvíjané pre potreby vyučovania Javy a OOP. BlueJ [9] je nástroj vyvíjaný na Deakinskej Univerzite [10] a Univerzite v Kente [11] s podporou firmy Sun Microsystems [12]. Prostredie je lokalizované do slovenského jazyka a umožňuje tvorbu plnohodnotných Java aplikácií a Java apletov [13]. Ukazuje sa, že prostredie je pre študentov ľahko zvládnuteľné, takže pozornosť je sústreďovaná na zvládnutie problematiky OOP.

3.1. Prostredie BlueJ

Na obrázku obr. 1. je znázornené hlavné okno aplikácie verzie 2.5.2 v slovenskej lokalizácii. V tomto okne študenti vytvárajú manažujú svoje projekty, vkladajú do nich nové triedy, môžu vidieť graficky znázornené závislosti medzi nimi (obrázok obr. 5.), otvárať ďalšie okná: editor zdrojového kódu (obrázok obr. 5.), terminál (obrázok obr. 4.), nástroj na ladenie (obrázok obr. 3.)…



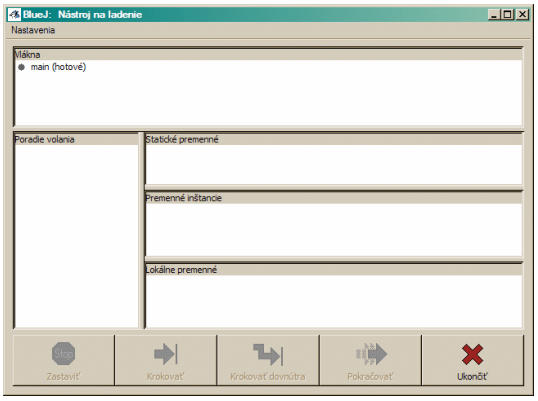
obr. 1. Hlavné okno BlueJ



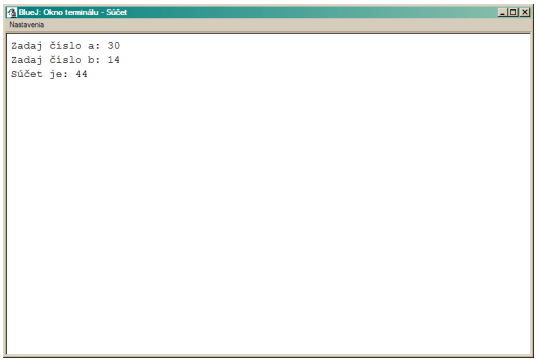
obr. 2. Editor BlueJ

Na obrázku obr. 5. je v zobrazení hlavného okna vidno vzájomné závislosti medzi triedami objektov. Šípky kategorizujú vzťahy do dvoch kategórií: dedičnosť a vzájomné

používanie objektov. Vďaka tomuto spôsobu zobrazenia majú študenti možnosť lepšie pochopiť vzájomné závislosti tried.

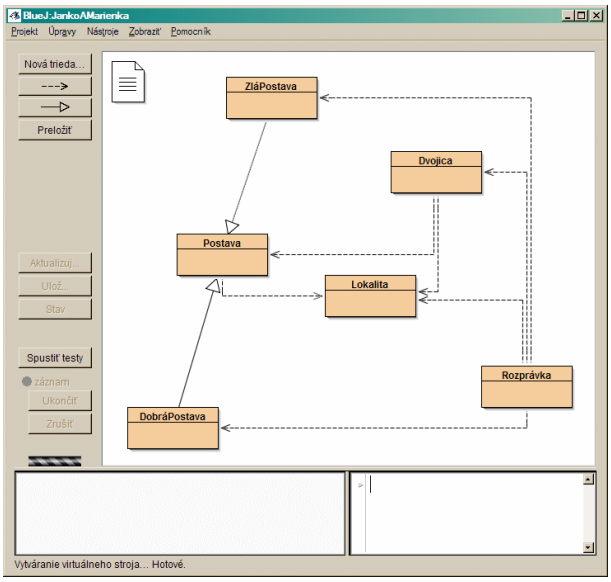


obr. 3. Okno nástroja pre ladenie v BlueJ



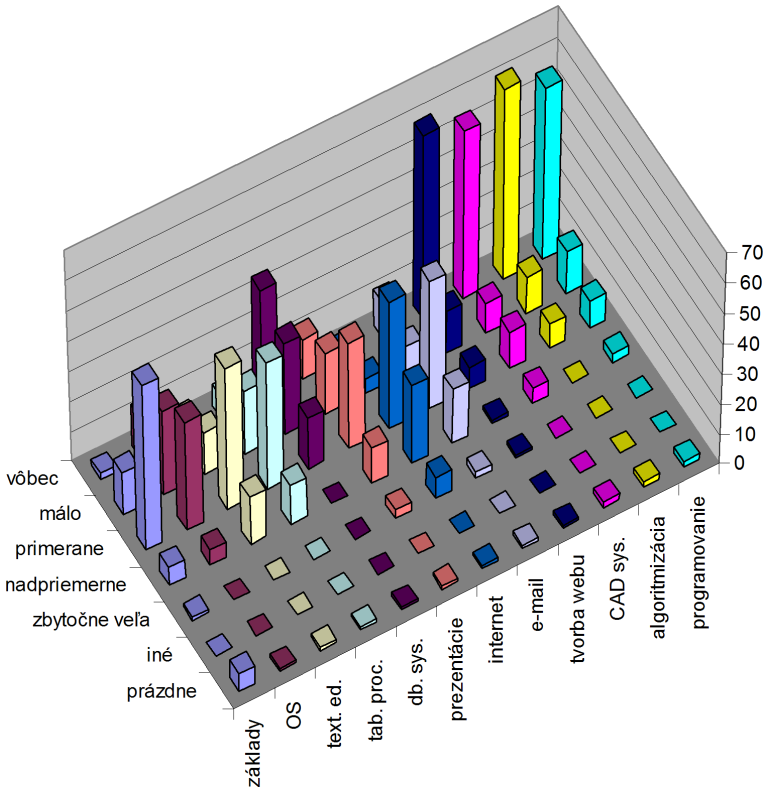
obr. 4. Terminál BlueJ

Obrázok obr. 3. znázorňuje okno nástroja na ladenie. Po preložení triedy bez chýb, je možné definovať v jej zdrojovom kóde body prerušenia. V týchto bodoch nastáva počas vykonávania pseudo-kódu jazyka Java prerušenie behu programu a prechod do režimu ladenia. V okne nástroja na ladenie môžu študenti posúvať vykonávanie programu krok po kroku a sledovať meniace sa hodnoty premenných, poradie volania metód atď. Režim ladenia nemusí slúžiť len na hľadanie a opravu chýb v programe. V tomto prípade slúži aj na lepšie pochopenie fungovania programu a toho, ako je vykonávaný.



obr. 5. Grafická vizualizácia tried Java v prostredí BlueJ

Väčšina študentov prijatých do programu v kombinácii s informatikou sa s programovaním ani algoritmizáciou na hodinách informatiky na svojej strednej škole nestretla. Vyplýva to z prieskumu, ktorého čiastočné výsledky zobrazuje graf Graf 1.



Graf 1: Odpovede študentov na otázky ohľadom objemu času venovaného vymenovaným problematikám na hodinách informatiky počas štúdia na strednej škole (vzorka 80 študentov)

Na grafe vidno, že vzorka študentov prijímaná na PdF TU do programov v kombinácii s informatikou zväčša pochádza zo škôl, kde bola náplň predmetu informatika tvorená najmä výučbou aplikačných programov (jadro tvorí balík kancelárskych aplikácií) a práca s aplikáciami pre prehliadanie webu a prácu s elektronickou poštou. Programovanie, algoritmizácia, rovnako ako tvorba webu alebo CAD systémy, stoja v úzadí.

Na základe toho by sa dalo obrazne povedať, že pri vyučovaní programovania pri týchto študentoch staviame, takpovediac, na zelenej lúke. To nemusí byť na škodu, pretože podľa názoru niektorých autorov (napr. [14]), študenti, ktorí boli vzdelávaní alebo sa zaoberali imperatívnym programovaním samostatne, často prinášajú do tvorby objektovo-orientovaných programov zlozvyky (bad habbits).

4. Záver

OOP je momentálne prevládajúcou programovacou paradigmou vo svete imperatívneho programovania. Stručným opisom základných charakteristík OOP sa zaoberal tento článok. Článok zároveň zhodnotil výber jazyka a nástroja slúžiacich na vyučovanie programovania na PdF TU.

Literatúra

[1] *Object-Oriented Programming Concepts – The Java™ Tutorials.* [online] «http://java.sun.com/docs/books/tutorial/java/concepts/» [cit. 20.10.2009]

[2] *Object Oriented Programming.* [online] «http://c2.com/cgi/wiki?ObjectOrientedProgramming [cit. 20.10.2009]

[3] *Object Oriented Programming Tutorial.* [online] «http://www.aonaware.com/OOP1.htm» [cit. 20.10.2009]

[4] *Object Oriented Programming – Wikibooks, collection of open-content textbooks.* [online] «http://en.wikibooks.org/wiki/Object_Oriented_Programming» [cit. 20.10.2009]

[5] *Object-oriented programming – Wikipedia, the free encyclopedia.* [online] «http://en.wikipedia.org/wiki/Object-oriented_programming» [cit. 20.10.2009]

[6] Yank, K. *Object Oriented Concepts in Java.* [online] «http://articles.sitepoint.com/article/oriented-concepts-java-1» [cit. 20.10.2009]

[7] *Imperative programming – Wikipedia, the free encyclopedia.* [online] «http://en.wikipedia.org/wiki/Imperative_programming» [cit. 20.10.2009]

[8] *The Imperative Programming Paradigm.* [online] «http://cmpe.emu.edu.tr/aelci/Courses/D-318/D-318-Files/plbook/imperati.htm» [cit. 20.10.2009]

[9] *BlueJ – Teaching Java – Learning Java.* [online] «http://www.bluej.org/» [cit. 20.10.2009]

[10] *Deakin University.* [online] «http://www.deakin.edu.au/» [cit. 20.10.2009]

[11] *Computer Science at Kent – School of Computing – University of Kent.* [online] «http://www.cs.kent.ac.uk/» [cit. 20.10.2009]

[12] *Sun Microsystems.* [online] «http://www.sun.com/» [cit. 20.10.2009]

[13] *Java applet – Wikipedia, the free encyclopedia.* [online] «http://en.wikipedia.org/wiki/Java_applet» [cit. 20.10.2009]

[14] *Viera K. Prolux.* [online] «http://www.ccs.neu.edu/home/vkp/» [cit. 20.10.2009]