DUM č. 1 v sadě

35. Inf-11 Objektové programování v Greenfoot

Autor: Lukáš Rýdlo

Datum: 08.06.2014

Ročník: studenti semináře

Anotace DUMu: Úvodní informace k celé sadě, metodika. Simulace života zajíce v Greenfoot - seznámení s prostředím aplikace a jazykem.

evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Sada projektů pro výuku programování mírně pokročilých studentů

Úvod k sadě DUMů

Tato sada DUMů obsahuje tři části a v ní dva projekty vhodné pro mírně pokročilé studenty programování. Části popořadě obsahují projekt simulační hry „Život zajíce“ v jazyce Java a prostředí Greenfoot, druhá část obsahuje tutéž simulační hru v jazyce C s využitím grafické (herní) knihovny Allegro (a IDE Dev-C) a třetí část je věnována projektu bitmapového konvertoru v jazyce C pro příkazovou řádku.

Tyto tři značně nesourodé projekty mají dát možnost šikovnějším studentům programování srovnat různé přístupy a paradigmata i prostředí. Na rozdíl od ustrnulé individuální a systematické výuky jednoho konkrétního jazyka je mají připravit na reálné programátorské úkoly v reálném prostředí, ve kterém programátor musí často volit různé přístupy nebo paradigmata a učit se vhodnější a pro daný úkol lépe použitelnější jazyk, dohledávat si potřebné informace ve zdrojích na internetu a nespoléhat jen na naučenou teorii.

V DUMech nebude vysvětlena teorie systematicky, ale zásadně pouze s orientací na řešení konkrétního problému a také formou odkazů na externí zdroje. Předpokládám, že studenti již ovládají základní syntax nějakého C-like jazyka a dovedou minimálně pracovat s proměnnými, podmínkami a (vnořenými) cykly, že dovedou sami a bez pomoci řešit jednoduché úkoly, umí program zkompilovat a spustit v nějakém IDE. Je vhodné mít probrané základy OOP, ale není to nutné.

První dva projekty (DUMy 1–7 a 8–14) mají totožné zadání, ale první je řešen v objektovém jazyce a druhý v jazyce strukturovaném (imperativním). Smysl zdvojeného řešení je předvést, jak se liší řešení téhož problému v různých jazycích a paradigmatech. Je možné pracovat paralelně na obou projektech, jako vhodnější se mi ale jeví nejprve kompletně vyřešit projekt objektově v prostředí Greenfoot a teprve poté kompletně v jazyce C (vzhledem k náročnosti). Pomůže to studentům nedělat v takové míře chyby spočívající v podobnosti jazyků. Je samozřejmě možné vynecháním konkrétní kapitoly ochudit finální řešení o některou funkcionalitu.

Třetí projekt je zamýšlen jako demonstrace multiplatformní aplikace pro textovou konzoli. Studenti by měli programovat v obyčejném textovém editoru (nikoliv IDE), aby si uvědomili souvislost čistého textu a kódu programu a také aby vnímali proces kompilace od vytvoření textového souboru přes překlad a linkování až po spuštění binárky. V prostředí Windows může být vhodnou volbou například PSPad nebo Notepad+. Doporučuji ale přimět studenty programovat přímo na příkazové řádce v Linuxu. K tomu lze využít LiveCD některé z distribucí, které obsahují kompilátor gcc a některý textový editor (nic víc není potřeba), např. Knoppix – v době psaní tohoto textu byl aktulání obraz KNOPPIX_V7.0.5CD-2012-12-21-EN.iso a bylo možné jej stáhnout např. z http://gd.tuwien.ac.at/linux/knoppix/KNOPPIX_V7.0.5CD-2012-12-21-EN.iso, lze vypálit a spustit z CD nebo v programu VirtualBox (další viz http://www.livecdlist.com/). Osobně preferuji editor Vim, ale lze použít také Emacs, Nano apod. Pro studenty může být jejich použití (obzvláště Vim) značně zdržující a nepohodlné, protože jim chybí návyky pro práci s editorem. Je proto možné nechat pomalejší studenty nebo složitější kapitoly psát například v grafickém editoru Gedit

(v Knoppixu je místo něj program Leafpad). Za stěžejní považuji studenty přimět zkompilovat kód jak v Linuxu, tak ve Windows, aby viděli rozdíl mezi binárkami pro jednotlivé systémy a uvědomili si význam multiplatformního zápisu kódu.

Upozornění k metodice a řešení

Samotné DUMy jsou určeny jako materiál pro učitele, nikoli studenty. Forma výuky může být různorodá: párové programování nebo programování jednotlivců či práce v týmu na dílčích úkolech. Jednotlivé DUMy tvoří ucelené bloky, které každý student zvládá svým vlastním tempem. Role vyučujícího je v usměrňování postupu studentů, příklady postrádají smysl, pokud bude učitel celé řešení demonstrovat nebo dělat za žáky. Učitel by také neměl vysvětlovat teorii, ale pouze nasměrovávat studenty na vhodné zdroje v internetu. V případě problému vysvětlovat pouze konkrétní chyby.

V každém DUMu jsou úkoly uvedeny vždy dvakrát, jednou jako podklad pro učitele včetně (neúplného) řešení (komentářů, návodů, poznámek) a poté bez řešení, takže tuto stránku stačí vytisknout a rozdat studentům, aby mohli pracovat samostatně a nemuseli se zdržovat čekáním na zadání.

V řešeních úkolů jsou zcela záměrně vynechávány některé řádky nebo zanášeny záměrné chyby, aby studenti, kteří DUMy objeví na internetu, nemohli použít celé řešení bez úvahy metodou copy-paste. Na takovéto záměrné chyby upozorním v textu. Kompletní a bezchybné řešení jsem ochoten poskytnout pouze v podobě binárky, nikoliv kódu, abych zamezil jeho opisování při domácích úkolech.

Poslední projekt bývá podobně zadáván jako úkol na FI MU a nerad bych kolegům kazil výuku uvedením plného řešení.

Projekt: Simulace života zajíce v Greenfoot – úvod

Úvod

První projekt je simulační hra v objektovém jazyce Java a prostředí Greenfoot. Studenty seznámíme se simulačními hrami na příkladu té nejjednodušší: Game of Life (hra života), kterou vymyslel John Conway. Jedná se o triviální „svět“ tvořený obvykle čtvercovou sítí políček, která mohou nebo nemusejí žít (znázorňuje se černou nebo bílou barvou) a znázorňují tak určitou populaci v daném místě. Hra má čtyři jednoduchá pravidla, podle kterých se v každém časovém kroku (čas je ve hře diskrétní, což znamená, že plyne po krocích, jako je ve hrách obvyklé) mění populace tak, že při přemnožení nebo příliš malém množství sousedů buňka umírá, při přiměřeném počtu sousedů setrvává při životě a je-li mrtvá a v okolí je správný počet živých buněk (rodičů), opět oživne. Pro demonstraci nalezneme celou řadu implementací, třeba http://pmav.eu/stuff/javascript-game-of-life-v3.1.1/ nebo http://www.julianpulgarin.com/canvaslife/.

Hra života sama o sobě je pěkným programátorským úkolem a lze ji použít také k demonstraci odděleného naprogramování herní logiky a „prezentační“ (grafické) vrstvy, protože zobrazení lze provádět v konzoli vypisováním pole např. hvězdiček, v grafice pak třeba jednoduchým vyplňováním okna barevnými čtverci, ale lze také naprogramovat zápis do bmp souboru (viz poslední projekt této sady DUMů).

Dalším příkladem simulačních her mohou být simulace kolonií mravenců a jejich hledání potravy či právě simulace zajíců. Na webových stránkách www.greenfoot.org lze v sekci Scenarios hledat projekty „ant“ nebo „rabbit“ a objevit cizí hotová řešení těchto her. Pěkné příklady pro demonstraci jsou http://www.greenfoot.org/scenarios/1016, http://www.greenfoot.org/scenarios/2246 (lze stáhnout a editovat kód), http://www.greenfoot.org/scenarios/3902 (simulace zajíců a lišek na čtvercové síti i se stáhnutelným kódem a grafem vývoje populace).

Příprava a prostředí

Studenti nechť si nainstalují Greenfoot ze stránek http://www.greenfoot.org/download – k dispozici je verze pro Windows, Linux i Mac, javovský jar archiv i verze spustitelná na Windows bez instalace z flashdisku, což je jediná verze, která nevyžaduje na počítač nejprve nainstalovat běhové a vývojové prostředí Java JDK. Varianta pro Android zatím neexistuje, jelikož je nutné přeprogramovat běhové prostředí a základní třídy a vyřešit rozdílnost ovládání na dotykovém displeji (chování třídy MouseInfo). Nicméně projekt, který se pokouší pomocí IDE Eclipse a sady tříd umožnit kompilaci pro Android existuje v podobě Droidfoot:

http://www-is.informatik.uni-oldenburg.de/~dibo/teaching/pkjava/droidfoot/index.html.

Je vhodné, aby si studenti vyzkoušeli stáhnout a otevřít některé scénáře z webu (například výše doporučené). Ještě užitečnější bude projít si demonstrační scénář „wombat“ (medvěd chodící dokola kolem herního plánu a požírající listy). Ten dostáváme při spuštění Greenfootu volbou „Open tutorial and tutorial scenario“.

Scénáře je potřeba nejprve překompilovat (tlačítko je vpravo dole pod seznamem tříd). Spouští se tlačítkem „run“ ve spodní části okna. Studenti by se měli podívat na kód tříd v otevřeném scénáři

(dvojklikem na rámeček v seznamu tříd v pravé části okna). Vysvětlíme, že aplikace Greenfoot je sama o sobě programem v Javě, který vytváří běhové prostředí pro náš kód, kterým de facto tuto aplikaci rozvíjíme. Naše třídy (kód) jsou svým způsobem plugin pro Greenfoot, který je frameworkem – rámcovou aplikací, která usnadňuje programování vytvořením standardizovaného prostředí, ve kterém jsou k dispozici kromě samotného okna předprogramované třídy World (reprezentuje svět/herní plán, je abstraktní a proto je nutné ji implementovat vlastním kódem), Actor (také abstraktní, slouží jako společný předek všech objektů, které budeme vkládat do herního plánu), Greenfoot (třída interagující s prostředím aplikace), GreenfootImage (třída pro reprezentaci obrázků v Greenfoot), GreenfootSound (reprezentace zvuků) a MouseInfo (reprezentace pohybu myší). Studenty odkážeme na dokumentaci http://www.greenfoot.org/files/javadoc/ a dále jednotlivé třídy „neučíme“, můžeme pouze vysvětlit na pár konkrétních příkladech volání metod.

Necháme studenty pozměnit chování tříd v ukázkových scénářích a všechny dotazy na význam metod a implementaci nějaké funkce směřujeme na dokumentaci. Přímé odpovědi podáváme pouze k otázkám syntaxe Javy, pokud není možné odkázat na nějaký vhodný zdroj na internetu – vhodný seriál o Javě je na serveru www.linuxsoft.cz (http://www.linuxsoft.cz/article_list.php?id_kategory=192), byť je potřeba občas korigovat některé nevhodné praktiky – například v díle „Java (5) – Řízení programu“ jsou pro začátečníka nevhodné příklady podmínek:

```
int a=0;boolean ok;
if (a<=0 ) ok=true; // zda je menší
```

Je nutné studentům vysvětlit, že takový zápis je zbytečný a píšeme přímo:

```
int a=0;boolean ok;
ok = (a<=0); // zda je menší
```

Může být také vhodné vysvětlit, že vynechání složených závorek za podmínkou může způsobovat později chyby, budeme-li dovnitř těla podmínky přidávat další kód.

Úkoly s řešením

1. Vytvořte nový scénář a v něm podtřídu RabbitWorld třídy World. Ať je herní plán velký 800 px × 600 px a každé „políčko“ (rozlišovací jednotka pohybu) jednopixelové.

   Řešení je triviální: stačí přes pravé tlačítko nad „World“ zvolit „New subclass…“, dále napsat jméno světa (camel-case, první písmeno velké) a zvolit obrázek na pozadí (lze také přímo nakreslit – v řádku s tlačítkem „Import from file“ je zcela vlevo tlačítko s ozubeným kolečkem, kde je volba vytvoření nového obrázku pomocí v OS nastaveného grafického editoru). V samotném kódu třídy bude uvnitř konstruktoru volání konstruktoru předka (třídy World) s příslušnými parametry, tedy super(800, 600, 1).

2. Vytvořte třídu Rabbit jako potomka třídy Actor. Nechť se po spuštění scénáře pohybuje v každém kroku o 1 políčko. (Vložte do herního plánu alespoň jednu instanci.)

   Řešení: vytvoříme třídu podobně jako třídu RabbitWorld, pouze v kódu třídy se plní metoda „`act`“, která se vykonává s každým „tiknutím hodin aplikace“ u všech instancí potomků třídy Actor vložených ve světě. V této metodě bude „`this.move(1);`“ – voláme metodu (činnost) sebe sama (this je klíčové slovo označující právě běžící instanci) a předáváme počet polí o které

se má instance posunout. Toto vysvětlujeme jen tehdy, pokud studenti dosud nemají zkušenost s jazykem Java a objektovým programováním. Bylo by však vhodné, aby při řešení tohoto projektu už nějaké znalosti alespoň OOP (lépe Javy) měli.

3. Exportujte projekt jako JAR archiv. Spusťte jar archiv z příkazové řádky (nebo i kliknutím).

   V okně aplikace vpravo nahoře je tlačítko „Share“ a pod ním volba „Application“. Příkaz na příkazové řádce je „`java -jar soubor.jar`“ (Linux), resp. „`java.exe -jar soubor.jar`“.

4. Přejmenujte JAR soubor tak, aby měl koncovku ZIP. Rozbalte jako zip soubor a prohlédněte si strukturu balíčku. Upravte obrázek run.png a vše znovu zazipujte do souboru soubor2.zip. Přejmenujte na soubor.jar. Spusťte.

   JAR soubory jsou ZIP balíky s pevně danou strukturou. Soubor run.png je obrázek tlačítka pro spuštění v běhovém prostředí Greenfootu.

Úkoly

1. Vytvořte nový scénář a v něm podtřídu RabbitWorld třídy World. Ať je herní plán velký 800 px × 600 px a každé „políčko“ (rozlišovací jednotka pohybu) jednopixelové.
2. Vytvořte třídu Rabbit jako potomka třídy Actor. Nechť se po spuštění scénáře pohybuje v každém kroku o 1 políčko. (Vložte do herního plánu alespoň jednu instanci.)
3. Exportujte projekt jako JAR archiv. Spusťte jar archiv z příkazové řádky (nebo i kliknutím).
4. Přejmenujte JAR soubor tak, aby měl koncovku ZIP. Rozbalte jako zip soubor a prohlédněte si strukturu balíčku. Upravte obrázek run.png a vše znovu zazipujte do souboru soubor2.zip. Přejmenujte na soubor.jar. Spusťte.