TECHNOLÓGIE JAVA

NÁVODY NA CVIČENIA

Jaroslav Porubän, Michaela Bačíková, Dominik Lakatoš

Katedra počítačov a informatiky
Fakulta elektrotechniky a informatiky
Technická univerzita v Košiciach

ISBN 123-45-678-9101-1

NÁZOV: Technológie Java - Návody na cvičenia
AUTORI: Jaroslav Porubän, Michaela Bačíková, Dominik Lakatoš
VYDAVATEĽ: Technická univerzita v Košiciach
ROK: 2018
VYDANIE: prvé
NÁKLAD: 10 kusov
ROZSAH: 50 strán

ISBN 123-45-678-9101-1

Obsah

Predhovor

Predkladaný učebný text obsahuje návody na praktické precvičovanie poznatkov z oblasti ... Zvýšená pozornosť je venovaná aj ... Vzhľadom na svoje určenie si tento učebný text nekladie za cieľ vyčerpávajúce pokrytie tejto rozsiahlej a dynamicky sa rozvíjajúcej oblasti.

Tematicky je text členený do častí zodpovedajúcich jednotlivým cvičeniam. Prvé cvičenie je venované ...

Tento učebný text vznikol na základe podkladov využívaných pri cvičeniach z predmetu Technológie Java na FEI TU v Košiciach v školskom roku 2017/2018. Tematicky predstavuje výber okruhov, ktorý je určený nie len študentom bakalárskeho stupňa študijného programu Informatika, ale osloviť môže aj ďalších čitateľov, ktorí sa venujú problematike ...

Košice, február 2018 Autor

1. Cvičenie - Technológia Java

Ciele

1. Oboznámiť sa s organizáciou predmetu Technológie Java. (krok 1)
2. Rozlíšiť druhy Java platformy - Standard Edition (Java SE), Enterprise Edition (Java EE), Micro Edition (Java ME). (krok 2)
3. Naučiť sa vyhľadávať zdroje, oboznámiť sa so štruktúrou Java API a používať kurzový materiál na http://docs.oracle.com/. (krok 4, 5, 6)
4. Zvládnuť nainštalovať Java SE SDK, preložiť a spustiť program v jazyku Java v príkazovom riadku. (krok 3)
5. Rozlíšiť typy a význam certifikátov pre platformu Java. (krok 7)

Úvod

Predmet Technológie Java je predmetom zameraným na výučbu programovania v jazyku Java na platforme Java Standard Edition.

Postup

Krok č. 1

Oboznámte sa s organizáciou predmetu Technológie Java a podmienkami udelenia zápočtu. V prípade nejasnosti sa opýtajte cvičiaceho. **Na každom cvičení (počnúc 4) musí byť študent schopný prezentovať aktuálny stav prác na štúdii Minesweeper. Bez aktuálnych zdrojových textov je jeho účasť na cvičení zbytočná. Hlavný dôraz je kladený na cvičeniach na samostatnú prácu a konzultáciu s cvičiacim.**

Krok č. 2

Oboznámte sa s portálom http://www.oracle.com/technetwork/java, troma edíciami technológie Java (SE, EE, ME), zistite spôsob inštalácie prostredia Java Standard Edition.

> **Úloha 0.1:** Aký je rozdiel medzi inštaláciou JDK, JRE a NetBeans?

Krok č. 3

Preložte a spustite program `HelloWorld`, zdrojový kód HelloWorld.java.

```
1  public class HelloWorld {
2    public static void main(String[] args) {
3      System.out.println("Hello World!");
4    }
5  }
```

Program preložte v príkazovom riadku použitím prekladača `javac`. Argumentom pre prekladač je **cesta k súboru s programom** (`HelloWorld.java`).

> **Poznámka:** Potreba rozlišovať veľké a malé písmená v ceste k súboru je závislá od platformy (MS Windows, Solaris, Linux, Mac OS).

```
c:\cvicenie01>javac HelloWorld.java

c:\cvicenie01>_
```

Obrázok 1: Preklad pomocou `javac`

Úloha 0.2: Zistite aký súbor pribudol po preklade v adresári. Čo obsahuje daný súbor?

Spustite program v prostredí JVM pomocou `java`. Argumentom pre JVM je **názov triedy** programu obsahujúcej metódu `main`.

> **Poznámka:** Vzhľadom na to, že Java rozlišuje veľké a malé písmená (podobne ako jazyk C, C++, C#), je nutné rozlišovať veľké a malé písmená pri zadaní mena triedy (`HelloWorld`).

```
c:\cvicenie01>javac HelloWorld.java

c:\cvicenie01>java HelloWorld
Hello World!

c:\cvicenie01>
```

Obrázok 2: Spustenie pomocou `java`

Úloha 0.3: Čo sa stane ak zadáte v príkazovom riadku`java helloworld`?

Úloha 0.4: Skúste premenovať v `HelloWorld.java` metódu `main` na metódu `start`. Podarilo sa program skompilovať a spustiť?

> **Poznámka:** Môžete použiť textový editor Notepad (Windows), Notepad++ (Windows) resp. jEdit (Solaris).

Krok č. 4

Oboznámte sa so štruktúrou inštalácie platformy Java - (Solaris: `/export/home/student/jdk1.8.0`, Windows: `c:\Program Files\Java\jdk1.8.0`).

Krok č. 5

Oboznámte sa s Java tutoriálom http://docs.oracle.com/javase/tutorial. Slovenský preklad vzniká na http://java.mrazovci.eu/.

Krok č. 6

Oboznámte sa s Java dokumentáciou http://docs.oracle.com/javase/8/docs/api – základná orientácia v Java API.

Krok č. 7

Oboznámte sa s Java Certification Path. Obsah predmetu Technológie Java je podobný (ale nie úplne totožný) požiadavkám a príprave na získanie certifikátu **Oracle Certified Professional Java Programmer, pôvodne Sun Certified Java Programmer**. Certifikácia OCPJP predpokladá aktívnu prax s technológiou Java v rozsahu minimálne 6 mesiacov. Viac informácií o certifikácií môžete získať na http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=140#13.

Úloha 0.5: Zistite kde je možné absolvovať certifikáciu OCPJP. Je nutné pred získaním certifikátu OCPJP Certified Java Associate?

Úloha 0.6: Koľko stojí certifikácia na OCPJP? Koľko stojí prípravný kurz na získanie certifikátu?

Zdroje

[1] The Java Tutorial http://docs.oracle.com/javase/tutorial
[2] J2SE reference documentation http://docs.oracle.com/javase/8/docs/api

Doplňujúce zdroje

[1] Cay S. Horstmann, Gary Cornell: Core Java, Volume 1: Fundamentals, 9th Edition. Prentice Hall PTR, December 2012, 1008 pp. ISBN 0137081898.
[2] Cay S. Horstmann, Gary Cornell: Core Java, Volume 2: Advanced Features, 9th Edition. Prentice Hall PTR, March 2013, 1152 pp. ISBN 013708160X.
[3] Pavel Herout: Učebnice jazyka Java, 5. vyd. Kopp, 386 pp. ISBN: 978-80-7232-398-2.

2. Cvičenie - IDE NetBeans

Ciele

1. Oboznámiť sa s prostredím NetBeans IDE https://netbeans.org. (krok 1)
2. Naučiť sa vytvoriť projekt, triedu v prostredí NetBeans IDE. (krok 1, 2, 3)
3. Oboznámiť sa s prostredím NetBeans IDE - dopĺňanie kódu, šablóny, komentovanie zdrojového kódu, ladenie programov. (krok 4, 5, 6, 8, 9)
4. Naučiť sa implementovať a ladiť jednoduché programy v prostredí NetBeans IDE. (krok 7, 10)
5. Vytvoriť projekt na školskom GitLab serveri za účelom priebežného odovzdávania zadania. (krok 12)

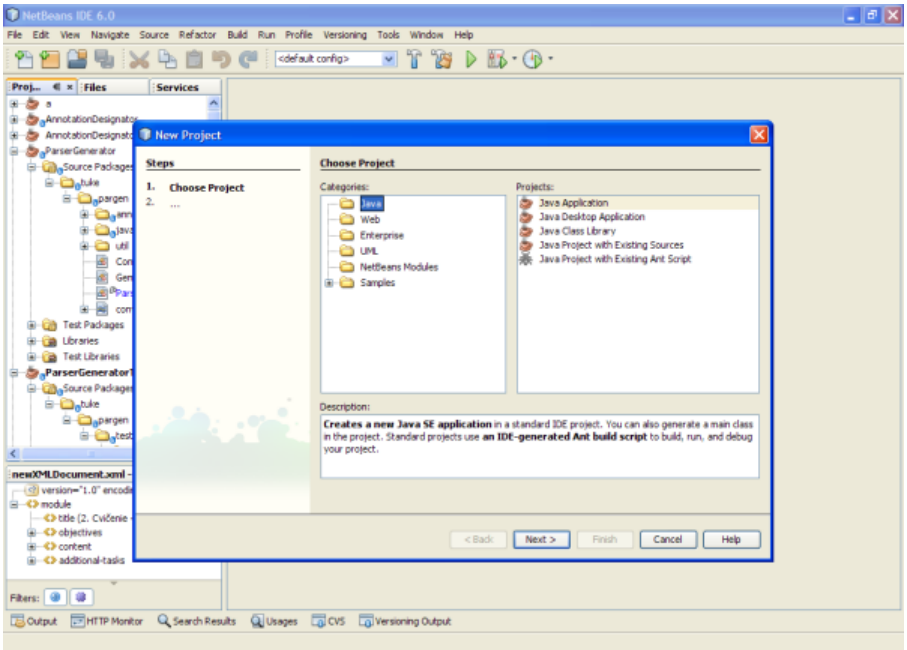
Úvod

V tomto cvičení sa budeme venovať práci v nástroji Netbeans IDE. Tiež si pripravíme základnú štruktúru projetktu na školskom GitLab serveri, kam budeme priebežne odovzdávať zadanie.

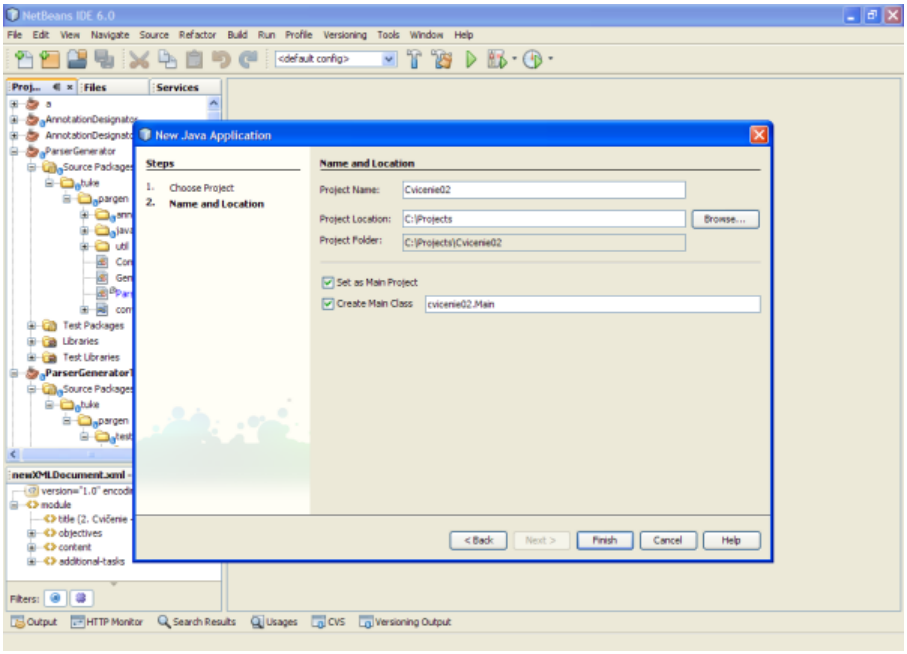
Postup

Krok č. 8

Vytvorte nový projekt v prostredí NetBeans. Vykonajte výber v menu: „File > New Project".



Vyberte kategóriu „Java" a v nej typ projektu „Java Application". Stlačte tlačidlo „Next".

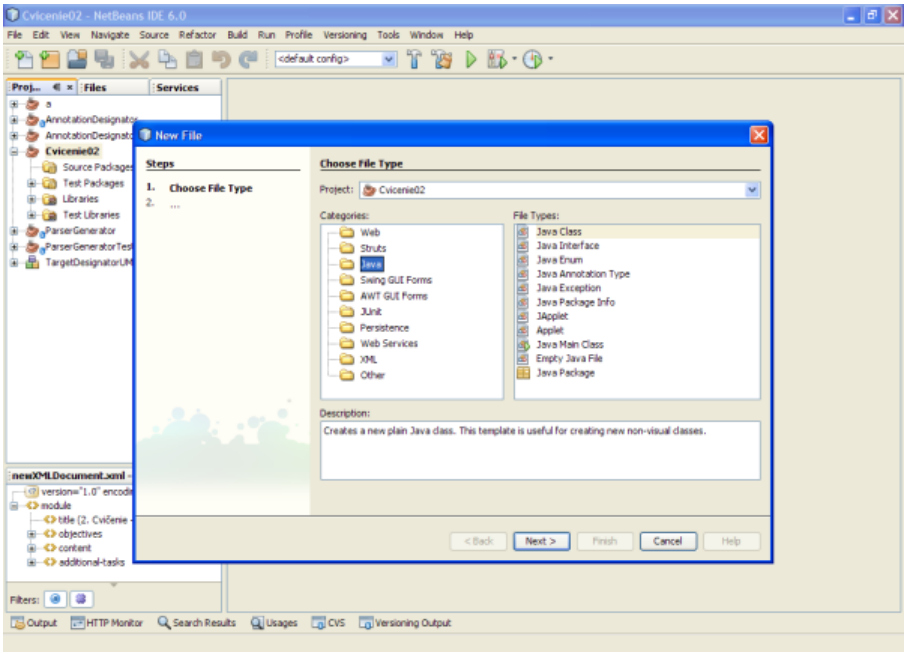


Napíšte názov projektu („Project Name”), napríklad `Cvicenie02`. Pre projekt sa vytvorí v súborovom systéme adresár s takým istým menom ako je názov projektu. Vyberte umiestnenie projektu („Project Location”). V textovom poli pri zvolení „Create Main Class” je možné zároveň s projektom vytvoriť hlavnú triedu projektu s určením jej názvu. Takúto triedu zatiaľ nevytvárajte. Taktiež je možné nastaviť vytváraný projekt ako hlavný (ak je vytvorených v prostredí viac projektov, tak jeden z nich je možné určiť ako hlavný) a to zvolením možnosti „Set as Main Project”. Po vyplnení požadovaných údajov stlačte tlačidlo „Finish”.

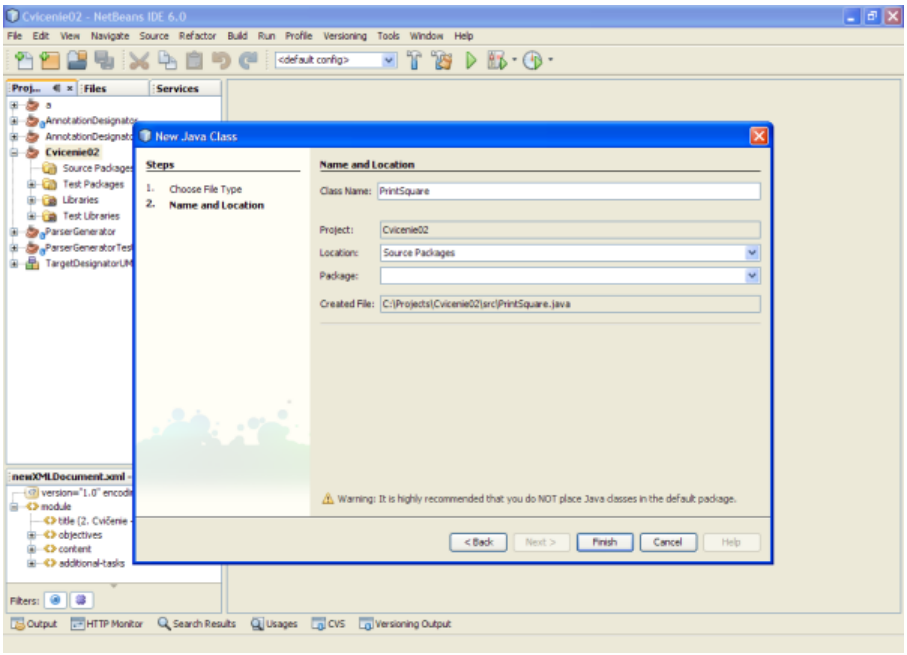
Poznámka: Súčasťou NetBeans IDE je aj pomoc (Help), ktorú je možné získať zvolením „Help > Help Contents” v hlavnom menu.

Krok č. 9

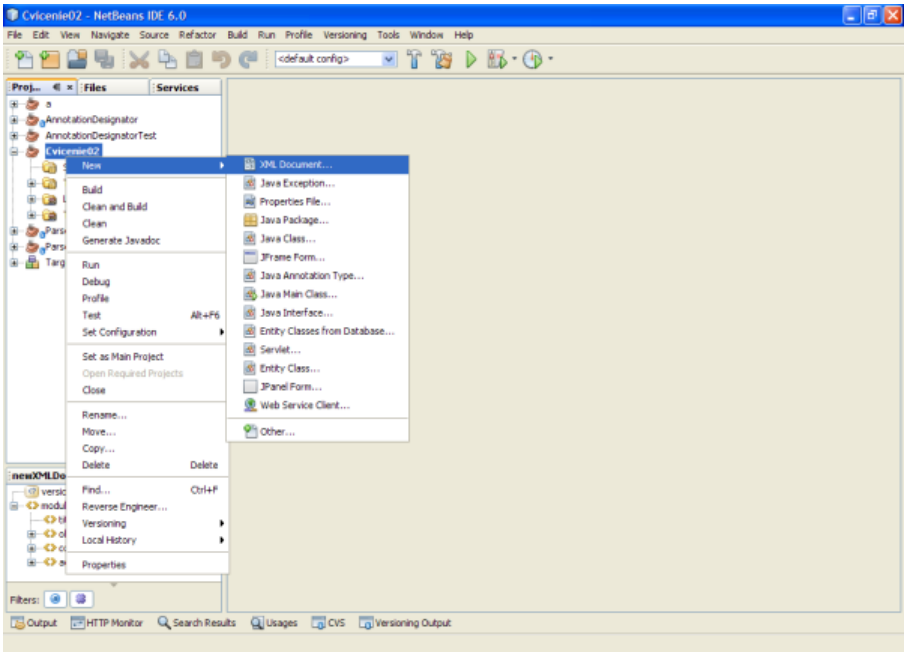
V tomto kroku pridáte do vytvoreného projektu súbor. Vyberte v menu položku „File > New File”. Vyberte kategóriu „Java” a v nej typ súboru „Java Class”. Stlačte tlačidlo „Next”.



Napíšte názov triedy (napríklad `PrintSquare`) do textového poľa „Class Name” a stlačte tlačidlo „Finish”.

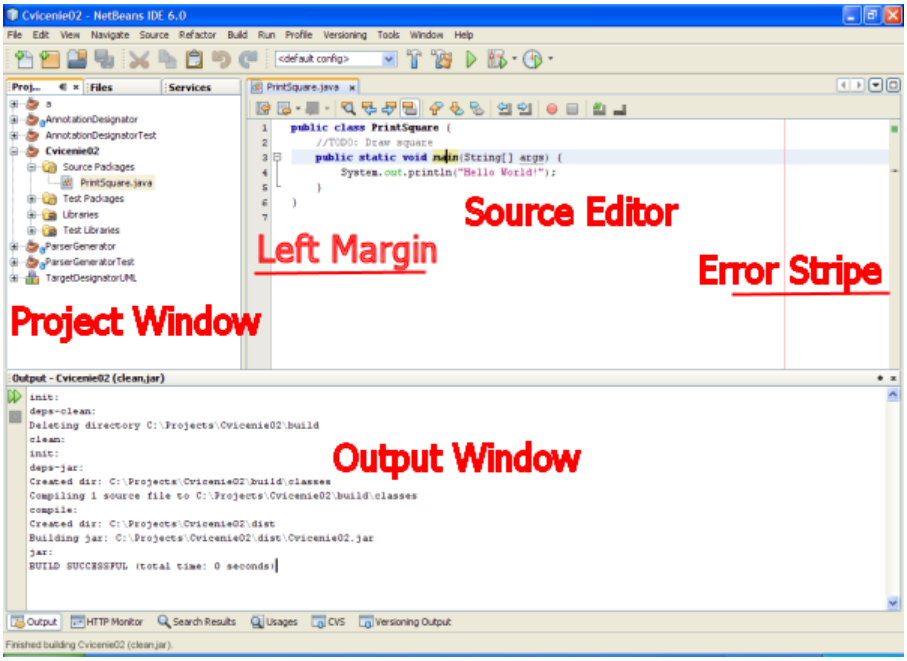


Poznámka: K dialógovému oknu pre vytvorenie triedy je sa možné dostať taktiež zo štruktúry záložiek vytvoreného projektu prostredníctvom kontextového menu nad záložkou „Projects”.



Krok č. 10

Zoznámte sa s prostredím projektu.

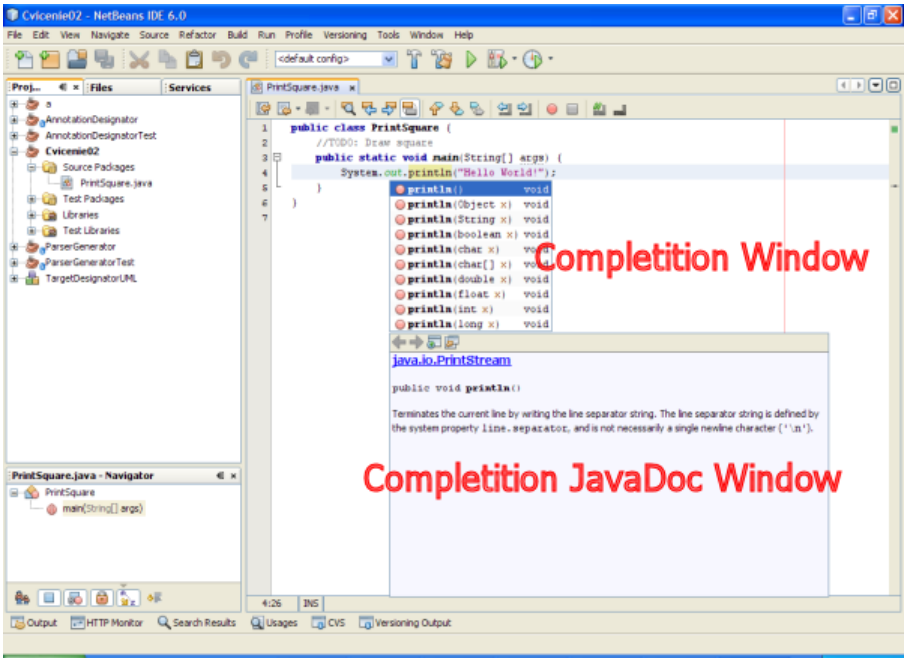


Úloha 0.7: Vytvorenú triedu `PrintSquare` upravte tak aby vypisovala `Hello World!`.

Úloha 0.8: Triedu preložte (menu „Build”) a spustite (menu „Run”).

Krok č. 11

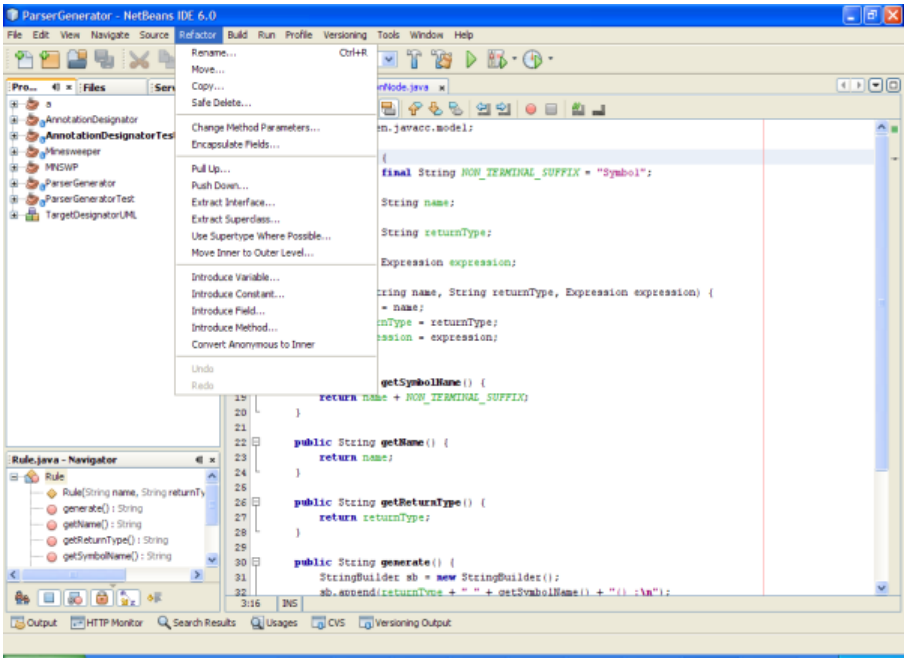
Oboznámte sa s automatickým doplňaním kódu (Code Completion) použitím skratky „CTRL+SPACE”, resp. „CTRL+\”. Môžete si to vyskúšať pri nastavení kurzora na triedu `System`, resp. metódou `println` v tele metódy `main` vytvoreného programu.



Krok č. 12

Oboznámte sa s refaktorizáciou programov v prostredí NetBeans. Refaktorizácia programov je proces zmeny štruktúry objektového programu bez zmeny jeho sémantiky. Medzi jednoduchú refaktorizáciu

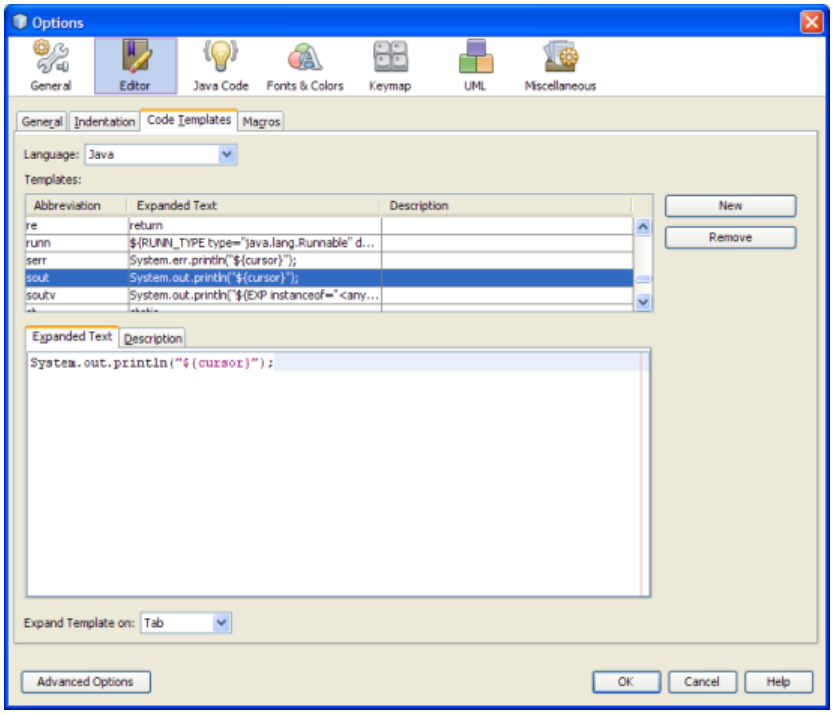
patrí napríklad premenovanie triedy (premennej, metódy). Použitie poloautomatickej refaktorizácie v prostredí NetBeans zjednodušuje dodatočné modifikácie zdrojového kódu.



Úloha 0.9: Premenujte vytvorenú triedu pomocou refaktorizácie „Rename” na `HelloWorld`.

Krok č. 13

Oboznámte sa s preddefinovanými šablónami a ich používaním prostredníctvom skratiek (Code Templates, Snippets) definovaných v nastaveniach prostredia NetBeans v menu „Tools > Options” voľba „Editor > Code Templates”.



Úloha 0.10: Otestujte použitie skratky `sout` vo vytvorenom programe. Napíšte súvisle `sout` a stlačte tabulátor (závisí od nastavenia v prostredí).

Úloha 0.11: Definujte vlastnú skratku `ma` pre vygenerovanie hlavnej metódy `main` programu v nasledujúcom tvare a otestujte ju.

```
1 public static void main(String[] args) {
2     ${cursor}
3 }
```

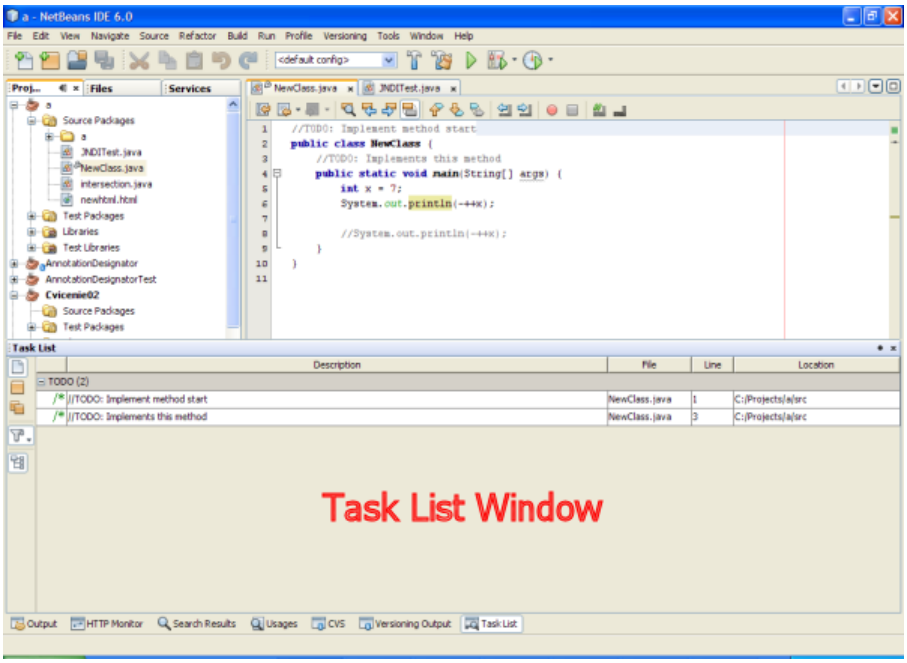
Krok č. 14

Pridajte do projektu súbor PrintTriangle.java. Tento súbor vložte do adresára `src` vytvoreného projektu. Tento program slúži na vykreslenie trojuholníka na štandardný výstup.

Úloha 0.12: Nájdite chyby v programe podľa pokynov v zdrojovom kóde PrintTriangle.java.

Krok č. 15

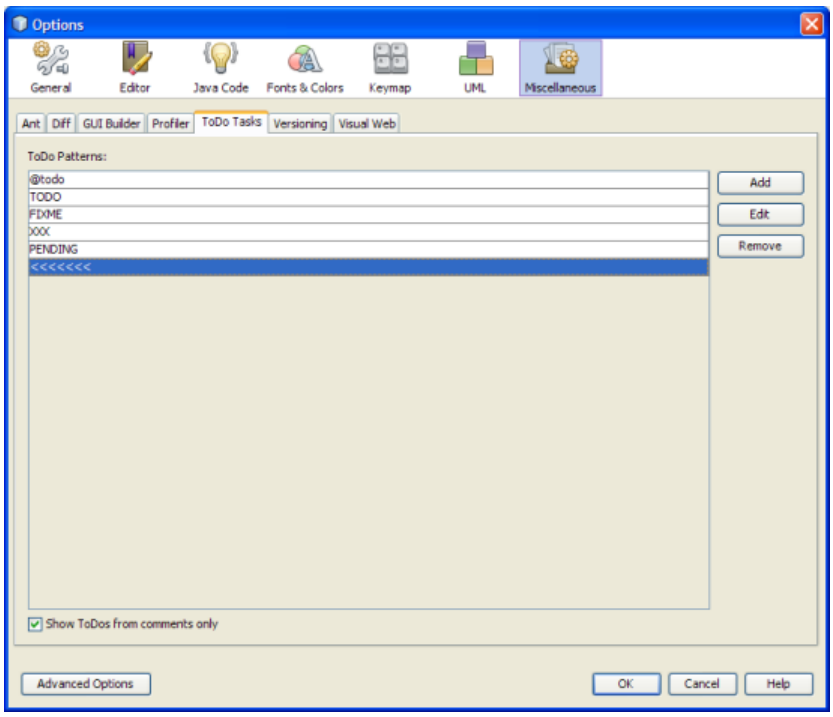
V prostredí Netbeans je možné zaznamenávať úlohy priamo v zdrojovom kóde. Zobraziť zoznam úloh je možné zobrazením okna „Task List" (v NetBeans 7.2 a vyššie „Action Items") z menu „Window".



Odskúšajte si zaznamenávať úlohy v zdrojovom kóde, ktoré je potrebné realizovať. Miesto realizácie úlohy sa označí prostredníctvom komentára a kľúčového slova, napríklad:

```
1 //TODO: Implement this method.
2 public static void main(String[] args) {
3 }
```

Kľúčové slová sú definované v nastavení „Tools > Options" voľba „Miscellaneous > ToDo Tasks".



Krok č. 16

Oboznámte sa s postupmi pre komentovanie zdrojového kódu. Nastavte kurzor jeden riadok nad metódu `main` a začnite písať dokumentačný komentár (`/**`). Po stlačení `Enter` prostredie vygeneruje automaticky kostru pre komentovanie zvolenej metódy.

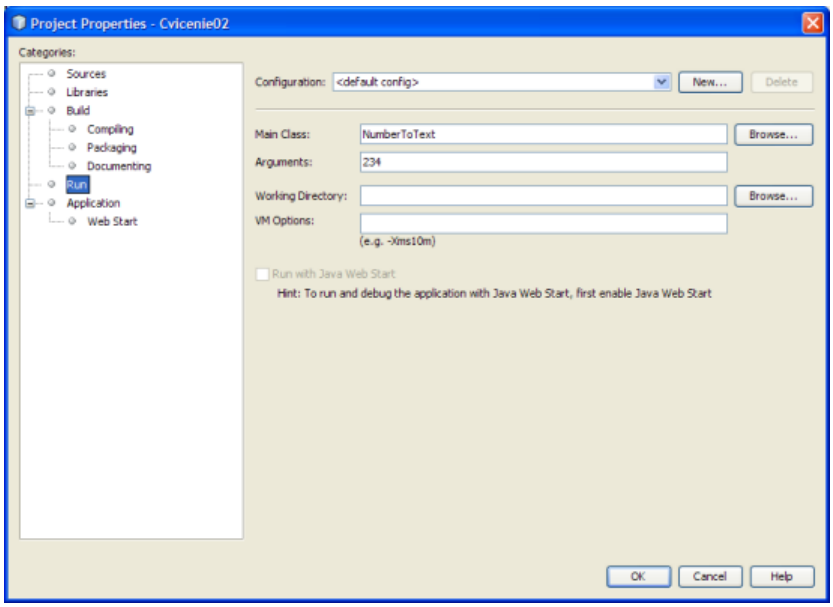
Úloha 0.13: Vygenerujte dokumentáciu zo zdrojového kódu projektu zvolením položky „Generate Javadoc" v kontextovom menu nad projektom, respektíve zvolením položky „Generate Javadoc for ..." v hlavnom menu „Build".

Krok č. 17

Pridajte do projektu program NumberToText.java.

Úloha 0.14: Nájdite chyby a upravte program NumberToText.java slúžiaci na prevod čísel na slovnú reprezentáciu.

Pre testovanie príkladu `NumberToText.java` nastavte túto triedu ako hlavnú a zadajte vstupný argument (testované vstupné číslo) v dialógu, ktorý získate výberom položky „Properties" v kontextovom menu nad projektom. V zobrazenom dialógovom okne vyberte voľbu „Run", nastavte hlavnú triedu po stlačení tlačidla „Browse" pre položku „MainClass" a zadajte hodnotu vstupného argumentu v položke „Arguments" podľa pokynov uvedených v zdrojovom kóde.



Krok č. 18

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Krok č. 19

Zadanie bude odovzdávané priebežne prostredníctvom systému na správu verzií Git na školskom GitLab serveri https://git.cnl.sk/. Kódovanie všetkých odovzdávaných súborov musí byť UTF8 alebo ASCII. Pri názvoch vytváraných súborov a priečinkov *záleží* na veľkosti písmen!

Úloha 0.15: Na školskom GitLab serveri vytvorte nový projekt s názvom **tjava-2015** a vykonajte push suboru `README` (bez prípony) s nasledujúcim obsahom:

```
STUDENT ID: S0123456789
GROUP : cisloSkupiny
```

kde `S0123456789` nahraďte za svoje jedinečné identifikačné čislo a `cisloSkupiny` nahraďte číslom Vašej skupiny v rozvrhovej jednotke. Oba údaje nastavte podľa systému MAIS - ID nájdete v študentskom rozhraní systému MAIS vo svojom profile. Číslo skupiny nájdete na http://maisportal.tuke.sk/portal/rozvrhy.mais.

Štruktúra Vášho projektu spolu s predpísaným súborom README bude v rámci cvičení nasledovná (štruktúru projektu Minesweeper zatiaľ nevytvárajte, stiahnete ho na nasledujúcom cvičení):

```
.
├── Minesweeper
│   ├── resources
│   ├── src
│   └── test
├── README
└── .gitignore
```

kde Minesweeper je hlavný adresár Vášho Java projektu.

Poznámka: Návody na prácu s GitLab-om a na nastavenie SSH kľúčov pre jeho používanie nájdete na http://it4kt.cnl.sk/c/pvjc/#/gitlab a http://it4kt.cnl.sk/c/pvjc/#/2015%252Flabs.02

Upozornenie: Upozornenie: Je dôležité, aby vaše súbory zachovali uvedenú štruktúru. Ak sa niektorý zo súborov síce v repozitári nachádza, ale v inom priečinku, bude to považované za chybu a takýto projekt nebude považovaný za správny.

Poznámka: V závislosti od toho, aké vývojové prostredie používate, sa vo vašom projekte môžu nachádzať aj priečinky vytvorené práve týmto prostredím (napr. projekt vytvorený v prostredí NetBeans bude obsahovať priečinok `nbproject`). Existencia týchto priečinkov v repozitári nebude považovaná pri hodnotení za chybu.

Zdroje

[1] Hlavná stránka NetBeans IDE https://netbeans.org

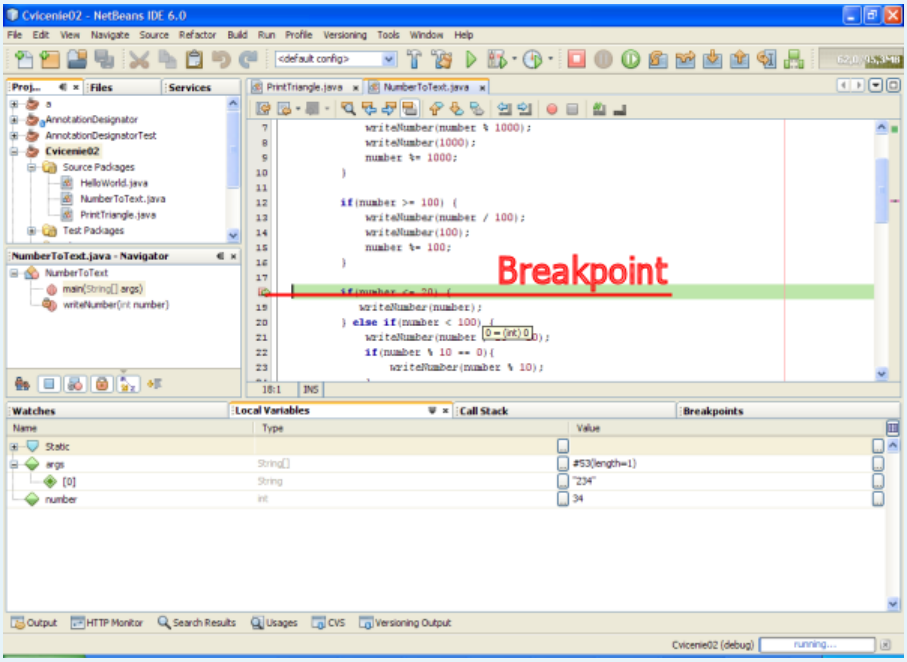
[2] Zdrojové kódy:

- PrintTriangle.java
- NumberToText.java

[3] Návody na prácu s GitLab-om, nastavenie SSH kľúčov: http://it4kt.cnl.sk/c/pvjc/#/gitlab a http://it4kt.cnl.sk/c/pvjc/#/2015%252Flabs.02.

Doplňujúce úlohy

Úloha 0.16: Otestujte podporu pre ladenie vytvorenej aplikácie spustením aplikácie v móde „Debug" výberom položky „Debug Main Project" v menu „Run".



Vložte prerušenie vykonávania programu (breakpoint) na ľubovoľný príkaz v metóde `main` a sledujte zmeny premenných počas vykonávania aplikácie v okne „Local Variables" a hodnoty

vami zvolených výrazov v okne „Watches” (výrazy pridáte prostredníctvom kontextového menu zobrazeného stlačením pravého tlačidla myši nad plochou okna „Watches”).

Cvičenie - Údajové typy (Minesweeper Task 1)

Ciele

1. Oboznámiť sa s podmienkami realizácie prípadovej štúdie Minesweeper. (krok 1, 2)
2. Naštudovať úvodnú špecifikáciu požiadaviek pre aplikáciu Minesweeper. (krok 3, 4, 5, 6, 7)
3. Rozlíšiť operátory jazyka Java, pochopiť ich význam a použitie.
4. Oboznámiť sa s významom a použitím objektov v programovaní.

Úvod

Na aktuálnom a nasledujúcich cvičeniach budete realizovať prípadovú štúdiu **Minesweeper**.

Postup

Krok č. 20

Návody na cvičenia budú obsahovať postupnosť krokov vedúcich k vytvoreniu aplikácie **Minesweeper** použitím technológie Java ale samotný program budete vytvárať vy. V štúdii budete pokračovať priebežne, preto je nutné zabezpečiť si prenos zdrojových textov na nasledujúce cvičenie. Vytvorenú aplikáciu **Minesweeper** je nutné odovzdať najneskôr v zápočtovom týždni. Odovzdávaná aplikácia musí obsahovať všetky požadované vlastnosti a mať požadovanú štruktúru v zmysle návodu na cvičenia. Súčasťou aplikácie je aj dokumentácia vygenerovaná zo zdrojových kódov a unit testy.

Krok č. 21

Minesweeper je počítačová hra určená pre jedného hráča. Cieľom hry je vyčistenie mínového poľa od mín bez ich detonácie.

Obrázok 3: Hra **Minesweeper**

Požiadavka klienta je vytvorenie hry **Minesweeper**. Detailnejšie informácie o hre je možné získať na www.wikipedia.org. Našťastie s niektorými úlohami vám pomáha skúsený spolupracovník a aj preto nebudete začínať na zelenej lúke. Napríklad v tomto cvičení vám spolupracovník dodal diagram tried, zopár zdrojových textov a niekoľko nápadov.

> **Poznámka:** Samozrejme, že v úlohe spolupracovníka vystupuje cvičiaci. (Aby ste vedeli koho sa pýtať a komu prejaviť vďaku:)

Krok č. 22

Vytvorte nový projekt v prostredí NetBeans IDE s názvom `Minesweeper`. Pridajte do adresára `src` vytvoreného projektu obsah archívneho súboru minesweeper.zip. Nastavte hlavnú triedu projektu v prostredí NetBeans na `minesweeper.Minesweeper`. Ak spustíte vytvorený projekt, dôjde k vzniku výnimky (`java.lang.UnsupportedOperationException: Method generateField not yet implemented`). Napriek tomu, že projekt je možné preložiť, hra nebude funkčná. Implementovať hru bude vašou úlohou.

Krok č. 23

Oboznámte sa s balíkmi v projekte.

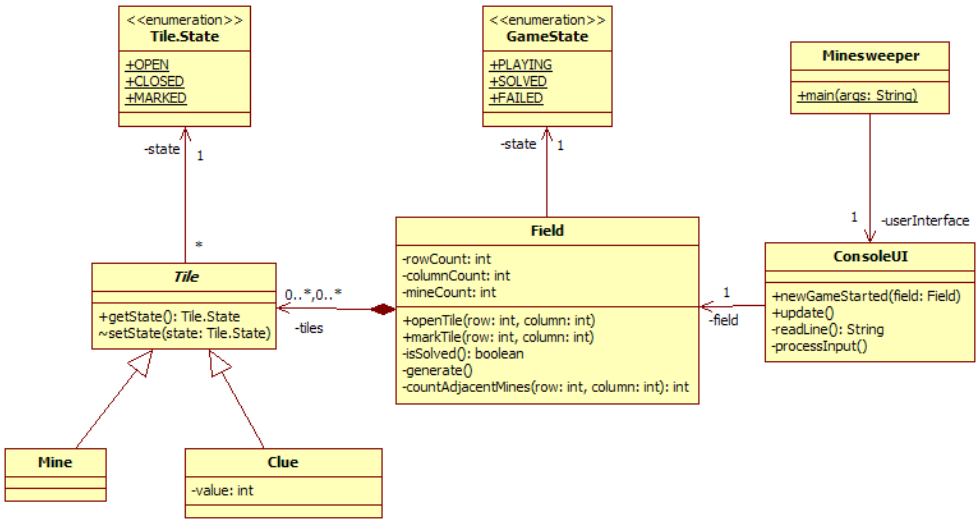
- Balík `minesweeper` je základným balíkom a obsahuje hlavné triedy aplikácie.
- Balík `minesweeper.core` obsahuje triedy definujúce logiku hracieho poľa nezávisle od používateľského rozhrania.
- Balík `minesweeper.consoleui` obsahuje triedy definujúce interakciu s používateľom.

Krok č. 24

Oboznámte sa s triedami v projekte, ich hierarchiou a väzbou.

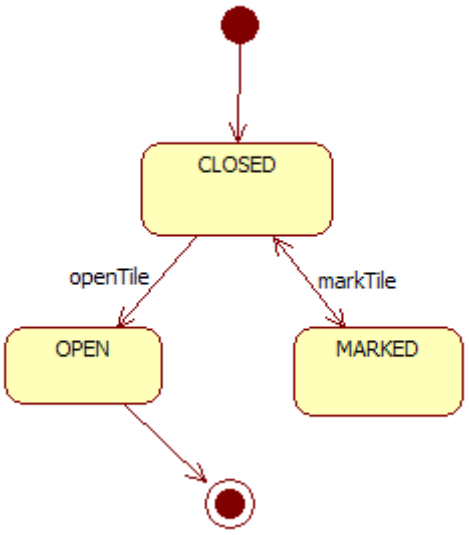
- Trieda `Minesweeper` je hlavnou triedou aplikácie, obsahuje metódu `main`.
- Trieda `ConsoleUI` definuje interakciu hracieho poľa s používateľom.
- Trieda `Field` reprezentuje hracie pole a jeho funkčnosť. Hracie pole obsahuje dlaždice.
- Trieda `Tile` reprezentuje dlaždicu hracieho poľa.
- Dlaždice sú dvoch typov: mína a pomoc pri hľadaní mín.
- Trieda `Mine` reprezentuje dlaždicu typu mína.
- Trieda `Clue` reprezentuje dlaždicu typu pomoc pri hľadaní mín.
- Stav dlaždice je vyjadrený pomocou enumerácie `Tile.State`.
- Stav hry je vyjadrený pomocou enumerácie `GameState`.

Nasledujúci obrázok je diagramom tried.



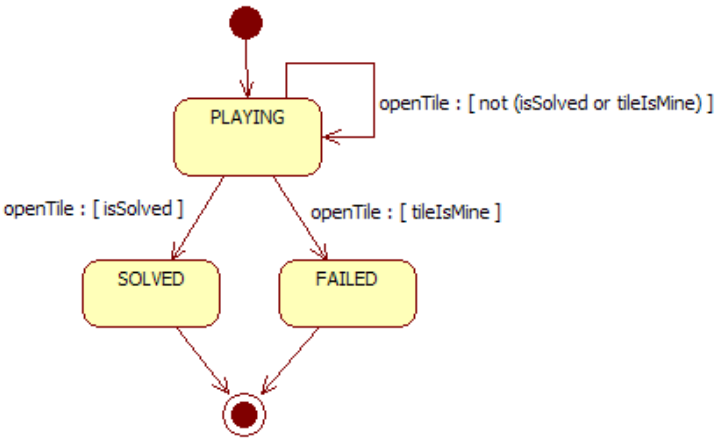
Obrázok 4: Diagram tried hry **Minesweeper**

Dlaždica môže byť v stave odkrytá `OPEN`, neodkrytá `CLOSED` alebo označená `MARKED`.



Obrázok 5: Diagram stavov dlaždice

Hra môže prebiehať (stav `PLAYING`), byť úspešne ukončená (stav `SOLVED`) alebo byť neúspešne ukončená (stav `FAILED`).



Obrázok 6: Diagram stavov hry

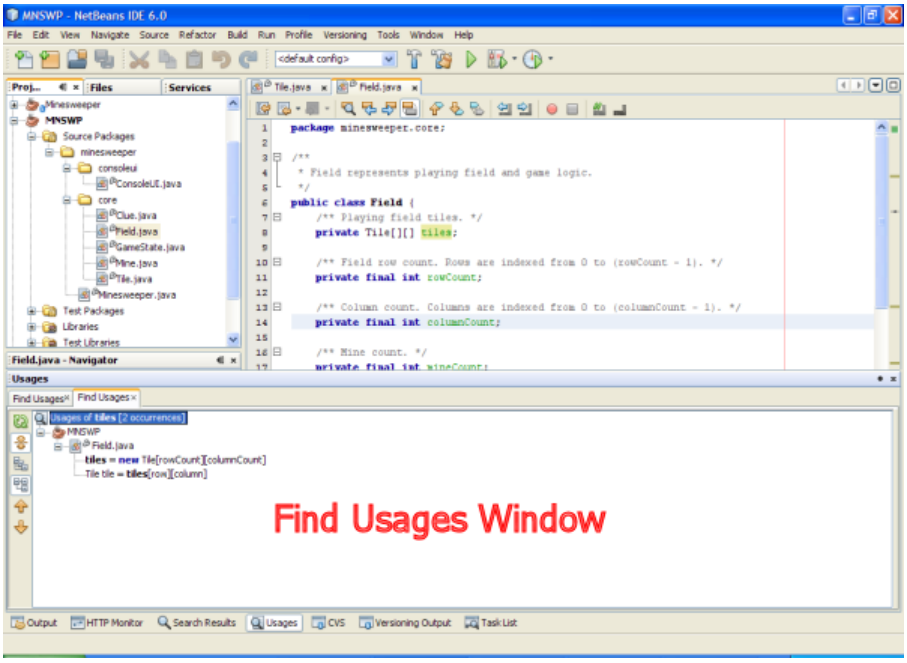
Krok č. 25

Vyskúšajte si prechádzanie medzi zdrojovými kódmi stlačením klávesy „CTRL" a súčasným stlačením ľavého tlačidla myši (mena triedy, premennej, metódy, atď.), resp. prostredníctvom kontextového menu nad hľadaným zdrojom zvolením položky „Navigate > ...".

Úloha 0.17: Otvorte v prostredí triedu `Field`. Prejdite pomocou „CTRL" a súčasným stlačením ľavého tlačidla myši na triedu `Tile` pomocou riadku `private Tile[][] tiles;`.

Krok č. 26

Naučte sa vyhľadávať miesta v zdrojovom kóde kde je použitá daná metóda, premenná resp. trieda. Nastavte kurzor na požadované miesto a zvoľte z kontextového menu „Find Usages".



Úloha 0.18: Vyhľadajte všetky miesta kde sa používa premenná tiles triedy `Field`.

Otvorte v prostredí triedu `Field`. Nastavte sa na deklaráciu členskej premennej `tiles` a zvoľte z kontextového menu „Find Usages".

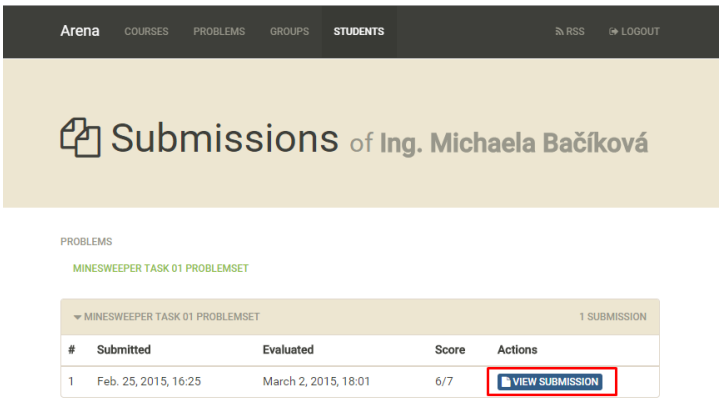
Úloha 0.19: Aké údajové typy su použité v danom projekte? Ktoré z nich sú primitívne?

Krok č. 27

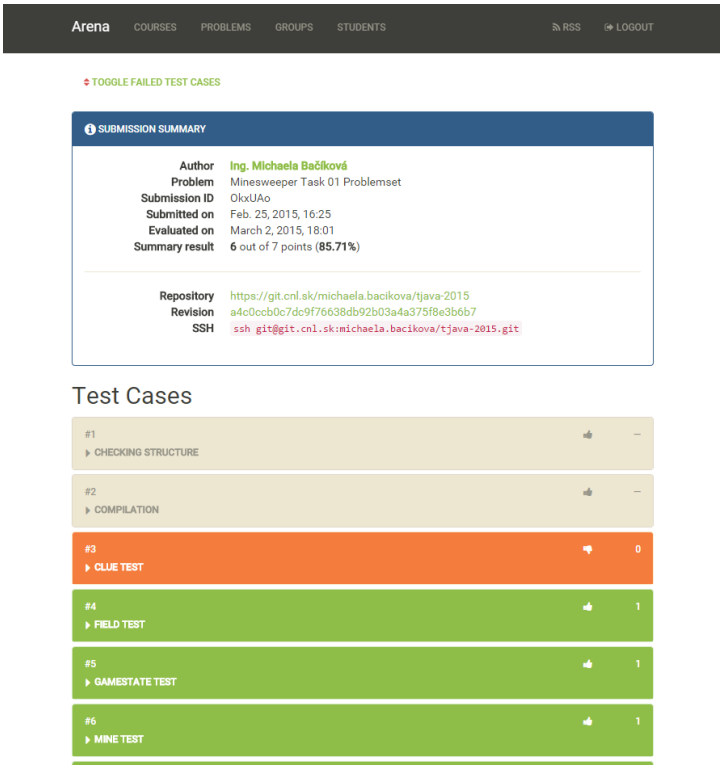
Pridajte vytvorený projekt do vášho projektu na serveri **GitHub**, ktorý ste vytvorili minulý týždeň. Štruktúra Vášho projektu spolu s predpísaným súborom README bude po dnešnom cvičení nasledovná:

```
.
├── Minesweeper
│   └── src
│       └── minesweeper
│           └── ...
├── README
└── .gitignore
```

Po úspešnom pridaní projektu do systému Git dostanete na Vašu študentskú e-mailovú adresu e-mail s názvom *"First submission of Minesweeper Task 01 Problemset"*. V jeho texte nájdete odkaz na vyhodnotenie v systéme **Arena**. Po kliknutí na tento odkaz by ste mali vidieť zoznam svojich odovzdaní:

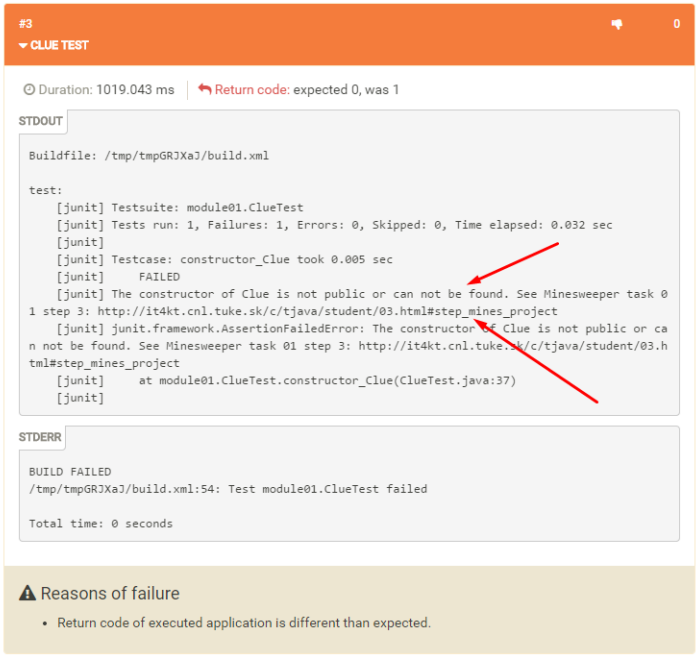


Nové odovzdanie sa zobrazí len ak bude vykonaný nový push do systému Git. Odovzdávania sa kontrolujú v intervale každých 3 hodín. Po kliknutí na "VIEW SUBMISSION" sa zobrazí konkrétne odovzdanie. V odovzdaní je možné vidieť zoznam jednotlivých testov, pričom zelenou farbou sú vyznačené úspešné testy a oranžovou neúspešné:



Po kliknutí na názov konkrétneho testu (napr. CLUE TEST) sa rozbalí detailný popis testu:

V prípade neúspešného testu je možné pri rozbalení detailu vidieť dôvod neúspechu a odkaz na modul a krok, v ktorom mala byť úloha vyriešená:



Úloha 0.20: Skontrolujte si svoje hodnotenie v systéme **Arena**.

Upozornenie: Zadanie musí byť odovzdané do systému Git vždy v danom týždni **najneskôr do 12:00 pred vaším cvičením**.

Krok č. 28

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Zdroje

[1] Detailnejšie informácie o hre **Minesweeper** www.wikipedia.org
[2] Zdrojové kódy: minesweeper.zip

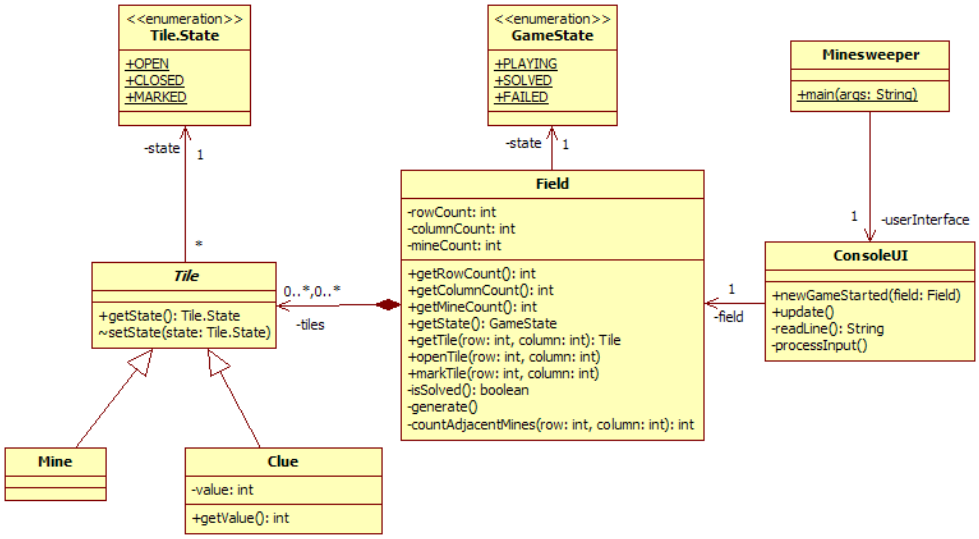
Objekty a triedy (Minesweeper Task 2)

Ciele

1. Naučiť sa vytvárať a používať vlastné balíky v jazyku Java.
2. Naučiť sa používať refaktorizáciu Encapsulate Fields v prostredí NetBeans IDE.
3. Vyskúšať si implementáciu objektových programov.

Úvod

V modeli sú uvedené metódy, ktoré nie sú uvedené v zdrojových kódoch. Úlohou dnešného cvičenia bude upraviť zdrojové kódy tak aby zodpovedali modelu.



Obrázok 7: Diagram tried aplikácie **Minesweeper** (na konci cvičenia 4)

Postup

Krok č. 29

Pri implementácii použite refaktorizáciu `Encapsulate Fields` v prostredí NetBeans IDE. Pomocou tejto refaktorizácie je možné pre zvolenú premennú objektu vygenerovať set respektíve get metódu - zapúzdrenie.

Úloha 0.21: Pridajte metódu int `getValue()` do triedy `Clue`.

Zvoľte členskú premennú `value` (nastavte kurzor na deklaráciu premennej value v triede `Clue`), stlačte pravé tlačidlo myši a z kontextového menu zvoľte „Refactor > Encapsulate Fields". Uistite sa, že je zvolené „Create Getter" a pokračujte pomocou tlačidla „Next".

Úloha 0.22: Pridajte metódy `int getRowCount()`, `int getColumnCount()`, `int getMineCount()` a `GameState getState()` do triedy `Field` použitím refaktorizácie `Encapsulate Fields`.

Úloha 0.23: Pridajte metódu `Tile getTile(int row, int column)` do triedy `Field`, ktorá vráti dlaždicu podľa zadaného riadku a stĺpca. Riadky a stĺpce sú číslované od `0`.

Úloha 0.24: Implementujte metódu `void markTile(int row, int column)` v triede `Field`. Metóda slúži na označenie/odznačenie dlaždice špecifikovanej riadkom a stĺpcom. V prípade, že je dlaždica neodkrytá (`CLOSED`) bude jej stav zmenený na označená (`MARKED`). Ak je dlaždica označená (`MARKED`) bude jej stav zmenený na neodkrytá (`CLOSED`). Riadky a stĺpce sú číslované od `0`.

Poznámka: Pri implementácii metódy `void markTile(int row, int column)` sa inšpirujte metódou `void openTile(int row, int column)`.

Krok č. 30

Ďalší krok pri implementácii hry **Minesweeper** je náhodné vygenerovanie obsahu herného poľa.

Úloha 0.25: Implementujte metódu `void generate()` v triede `Field` tak, aby v hernom poli tiles náhodne rozložila míny (`Mine`) a doplnila pomocné polia (`Clue`), pričom počet mín, ktoré majú byť rozložené je daný premennou `mineCount`.

Poznámka: Náhodné vygenerovanie hracieho poľa rozdeľte do dvoch fáz
- náhodné uloženie mín do poľa - metóda `void generateMines()`
- doplnenie pomocných polí na miesta, ktoré neobsahujú mínu - metóda `void fillWithClues()`

Inšpirujte sa nasledujúcim fragmentom zdrojového kódu.

```
1  private void generate() {
2      generateMines();
3      fillWithClues();
4  }
```

Poznámka: Pre náhodné generovanie čísel vytvorte objekt triedy `java.util.Random` (použite metódu objektu `int nextInt(int n)`).

Poznámka: Na náhodne zvolené súradnice v hernom poli vložte dlaždicu typu `Mine` ak pole na daných súradniciach už neobsahuje mínu (`tiles[row][column] == null`). Postup opakujte pokiaľ nebude uložený požadovaný počet mín. Po realizácii tejto úlohy bude hracia plocha obsahovať iba dlaždice typu mína.

Poznámka: Doplňte herné pole dlaždicami typu `Clue` (pomoc pri vyhľadávaní). Pre tento typ dlaždice je potrebné nastaviť hodnotu identifikujúcu počet dlaždíc s mínami v jej priamom susedstve.

Poznámka: Prejdite v cykle celé hracie pole. Pre každú neobsadenú pozíciu (platí `tiles[row][column] == null`) zistite počet susedných mín pomocou už implementovanej metódy `int countAdjacentMines(int row, int column)`.

Krok č. 31

Okomentujte všetky doposiaľ vytvorené metódy.

Krok č. 32

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Rozhrania (Minesweeper Task 3)

Ciele

1. Precvičiť vytváranie objektových programov v jazyku Java. (krok 1, 2)
2. Pochopiť význam rozhraní v objektovom programovaní. (krok 2, 3)
3. Naučiť sa vytvárať rozhrania v jazyku Java.
4. Implementovať generovanie hracej plochy pre hru Minesweeper a zobrazenie herného poľa.

Úvod

Jednou z požiadaviek na hru je možnosť používateľa vybrať si rozhranie pre hru (konzolové alebo grafické rozhranie Swing). Práve z tohto dôvodu je nutné zabezpečiť nezávislosť implementácie hernej logiky aplikácie od implementácie používateľského rozhrania. Dôvodom používania rozhraní je práve zabezpečenie implementačnej nezávislosti jednotlivých častí systému. Pre vytvorenie rozhrania, ktoré určí predpis pre pripojenie rôznych typov používateľského rozhrania použite postup uvedený nižšie.

Postup

Krok č. 33

Úloha 0.26: V balíku minesweeper definujte rozhranie `UserInterface` ako predpis, ktorý musia spĺňať používateľské rozhrania pre aplikáciu **Minesweeper**.

Toto rozhranie vygenerujte z existujúceho zdrojového kódu triedy ConsoleUI nasledujúcim postupom.

- Nastavte sa do zdrojového súboru triedy `ConsoleUI` priamo na meno tejto triedy.
- Stlačte pravé tlačidlo myši a z kontextového menu nad zdrojovým kódom zvoľte „Refactor > Extract Interface…".
- Do textového poľa zapíšte názov rozhrania `UserInterface`.
- Zvoľte metódy `newGameStarted` a `update`, ktoré majú byť zahrnuté vo vytváranom rozhraní a pokračujte ďalej.
- Presuňte vytvorené rozhranie `UserInterface` do balíka `minesweeper` použitím Drag&Drop v stromovej štruktúre projektu (záložka „Projects").
- V triede `Minesweeper` zmente typ členskej premennej `userInterface` z `ConsoleUI` na `UserInterface`.

Krok č. 34

V tomto kroku zobrazíme herné pole na štandardný výstup. Na zobrazovanie hracieho poľa na štandardný výstup sú kladené nasledujúce požiadavky:

- Riadky sú označované veľkými písmenami zoradenými podľa abecedy (A, B, ..., I).
- Stĺpce sú označované číslicami (0, 1, ... , 8).
- Pre vykreslenie odkrytej dlaždice (`OPEN`) typu mína (`Mine`) sa používa znak `X`.
- Pre vykreslenie dlaždice v stave (`OPEN`) typu pomoc pri hľadaní (`Clue`) sa používa číslo reprezentujúce hodnotu dlaždice.

- Pre vykreslenie označenej dlaždice (`MARKED`) sa používa znak `M`.
- Pre vykreslenie neoznačenej dlaždice (`CLOSED`) sa používa znak `-`.

```
   0  1  2  3  4  5  6  7  8
A  0  1  1  1  -  -  M  1  0
B  0  1  -  -  -  -  3  1  0
C  1  2  -  -  -  -  2  0  0
D  -  -  -  -  -  -  2  0  0
E  1  1  1  -  -  -  1  0  0
F  0  0  1  -  -  -  1  0  0
G  0  1  2  -  -  -  1  1  1
H  0  1  M  -  -  -  -  -  -
I  0  1  -  -  -  -  -  -  -
Please enter your selection (X) EXIT, (MA1) MARK, (OB4) OPEN :
```

Úloha 0.27: Implementujte metódu `void update()` v triede `ConsoleUI`, ktorá zabezpečí vykreslenie herného poľa.

Pre zobrazenie formátovaného výstupu (napríklad pri zobrazení označenia riadkov) použite metódu `System.out.printf(...)`.

Krok č. 35

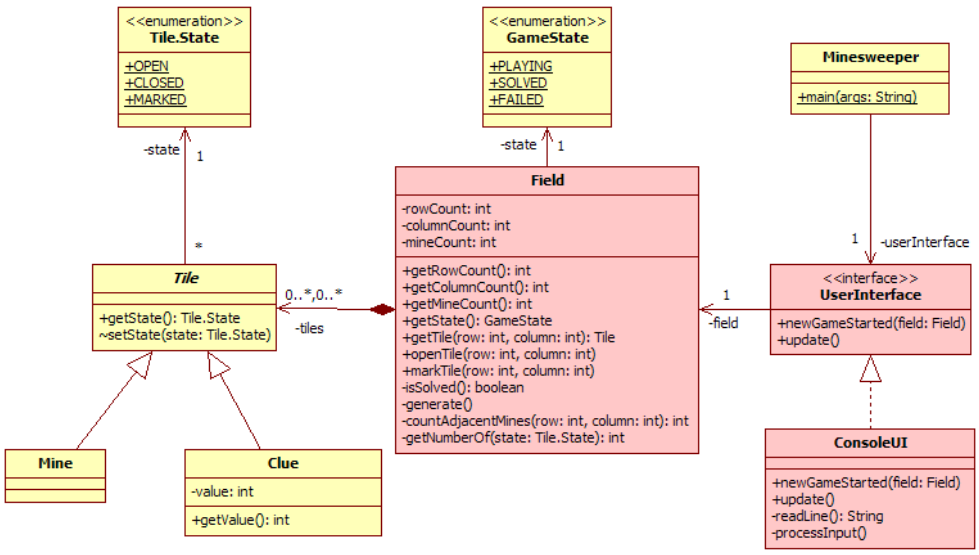
Pri každom odkrytí herného poľa používateľom sa testuje možnosť ukončenia hry (úspešne alebo neúspešne).

Úloha 0.28: Implementujte metódu `boolean isSolved()` definovanú v triede `Field`, ktorá určuje úspešné odkrytie hracieho poľa.

Hra je úspešne ukončená ak platí **počet všetkých dlaždíc - počet odokrytých dlaždíc = počet mín**. Z uvedeného vyplýva, že hra bude ukončená vtedy, ak budú odokryté všetky dlaždice bez mín. Pre určenie počtu odkrytých dlaždíc definujte súkromnú metódu `int getNumberOf(Tile.State state)`, ktorá vráti počet dlaždíc v danom stave.

Krok č. 36

Aktuálny model tried aplikácie je nasledujúcom obrázku.



Obrázok 8: Diagram tried aplikácie **Minesweeper** (na konci cvičenia 5)

Krok č. 37

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

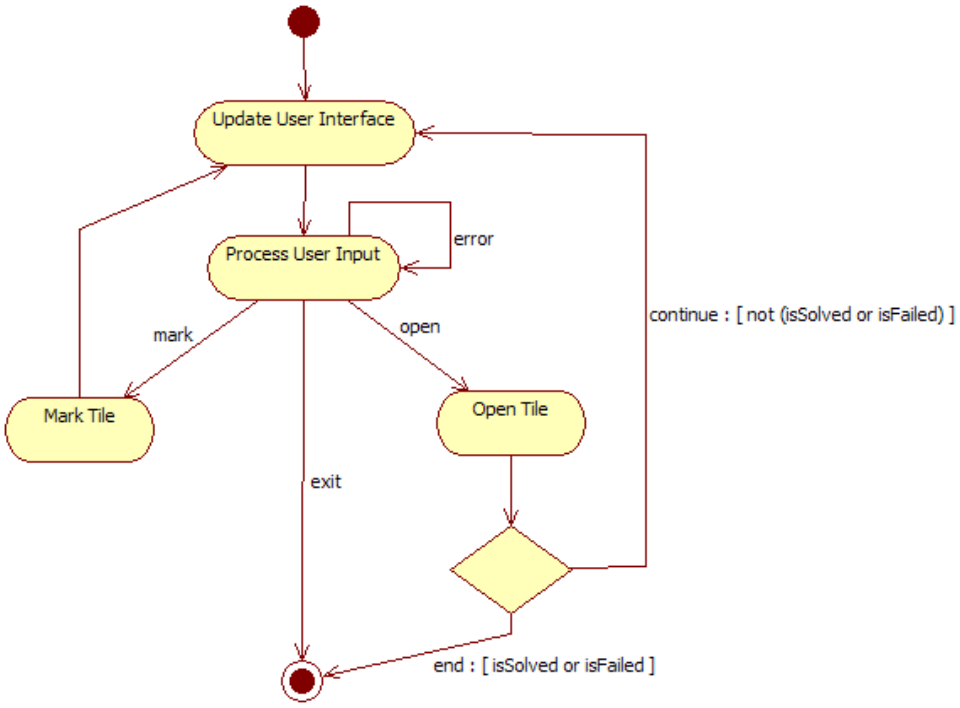
Interaktívne programy (Minesweeper Task 4)

Ciele

1. Oboznámiť sa s problematikou implementácie interaktívnych systémov. (krok 1, 2, 4)
2. Naučiť sa používať regulárne výrazy pre spracovanie vstupov. (krok 2, 3, 4)
3. Naučiť sa používať rekurzívne funkcie. (krok 4)

Úvod

Úlohou dnešného cvičenia je implementácia interakcie s hráčom. Pri interakcii je potrebné zabezpečiť vyžiadanie údajov od používateľa, overiť správnosť zadania načítaných údajov, zabezpečiť spätnú väzbu pre používateľa v prípade zle zadaných údajov a vykonať požadovanú operáciu.



Obrázok 9: Stavový diagram interakcie s hráčom

Postup

Krok č. 38

Úloha 0.29: Implementujte metódu `void processInput()` v triede `ConsoleUI` zabezpečujúcu interakciu s hráčom.

Metóda `void processInput()` má:

- Vypísať požiadavku na zadanie vstupu so vzorom očakávaného vstupu od používateľa: `X` – ukončenie hry, `MA1` – označenie dlaždice v riadku A a stĺpci 1, `OB4` – odkrytie dlaždice v riadku B a stĺpci 4.
- Načítať vstupnú požiadavky od používateľa (metóda `String readLine()`).
- Overiť správnosť vstupného reťazca. Na overenie správnosti vstupu použite regulárne výrazy.
- V prípade požiadavky od používateľa, ktorá nie je v požadovanom tvare je potrebné vyžiadať od používateľa opätovné zadanie vstupu.
- Na základe identifikácie akcie od používateľa vykonať požadovanú operáciu (označenie dlaždice, odkrytie dlaždice, ukončenie programu).

Poznámka: Vytvorte objekt triedy Pattern na definíciu vzoru vstupu pomocou regulárneho výrazu (statická metóda `Pattern.compile("O([A-I])([0-8])")`). Pre zadaný vstup overte konzistenciu so vzorom pomocou objektu triedy `Matcher`. Objekt triedy `Matcher` je možné získať pomocou metódy `Matcher matcher(CharSequence input)` vytvoreného objektu `Pattern`. Na overenie vstupov použite metódy `boolean matches()` a `String group(int group)` definované v triede `Matcher`.

Krok č. 39

Ďalším krokom je doplnenie reakcie na úspešné/neúspešné ukončenie hry.

Úloha 0.30: Doplňte do metódy `void newGameStarted(Field field)` v triede `ConsoleUI` test na úspešné (`field.getState() == GameState.SOLVED`) resp. neúspešné (`field.getState() == GameState.FAILED`) ukončenie hry.

Poznámka: V prípade úspešného resp. neúspešného ukončenia vypíšte oznam o ukončení a ukončite hru pomocou `System.exit(0)`.

Krok č. 40

Jednou z funkcií hry **Minesweeper** je automatické odkrytie všetkých susedných dlaždíc v prípade odkrytia dlaždice typu `Clue`, ktorej hodnota je rovná `0` (žiadna zo susedných dlaždíc nie je mína).

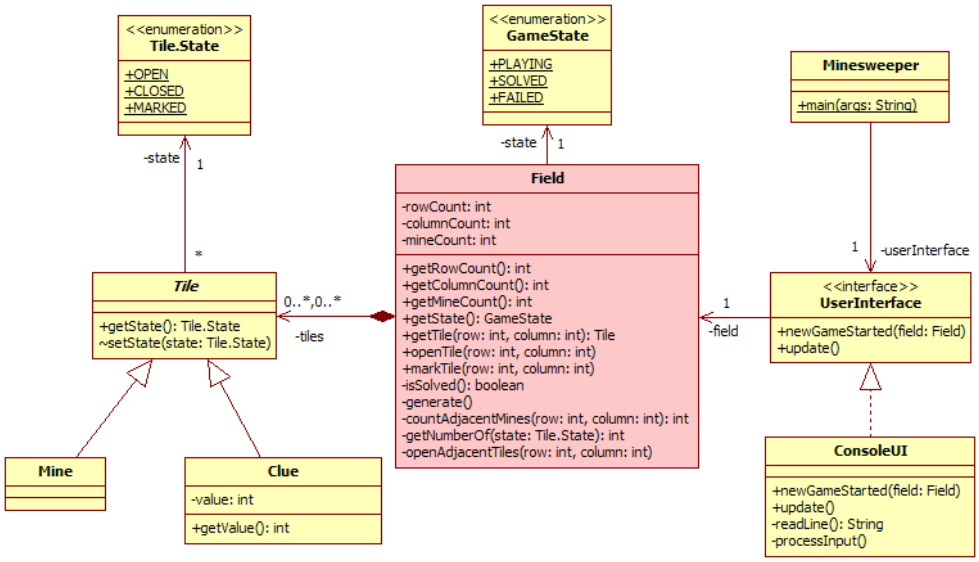
Úloha 0.31: Implementujte súkromnú metódu `void openAdjacentTiles(int row, int column)` v triede `Field` zabezpečujúcu odkrytie všetkých susedných dlaždíc. V tejto metóde použite na odkrytie susedných dlaždíc metódu `void openTile(int row, int column)`.

Poznámka: Pri implementácii metódy nezabudnite na prípady keď je dlaždica na okraji/v rohu hracieho poľa.

Úloha 0.32: Doplňte do metódy `void openTile(int row, int column)` aplikovanie metódy `void openAdjacentTiles(int row, int column)` v prípade odkrytia dlaždice typu `Clue` s hodnotou `0`.

Krok č. 41

Aktuálny model tried aplikácie je nasledujúcom obrázku.



Obrázok 10: Diagram tried aplikácie Minesweeper (na konci cvičenia 6)

Krok č. 42

Otestujte funkčnosť a použiteľnosť hry **Minesweeper**.

Krok č. 43

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Doplňujúce úlohy

Úloha 0.33: Akceptujte určenie riadku s malým alebo veľkým písmenom vo vstupnej požiadavke ako aj určenie akcie pre označenie dlaždice – `M/m`, resp. odokrytie dlaždice – `O/o`.

Výnimky a testovanie programov (Minesweeper Task 5)

Ciele

1. Oboznámiť sa s významom a druhmi výnimiek v jazyku Java. (krok 1, 2)
2. Naučiť sa vytvárať a používať triedy výnimiek definované programátorom. (krok 2, 3)
3. Naučiť sa vytvárať unit testy pre aplikáciu pomocou NetBeans IDE.

Úvod

Úlohou dnešného cvičenia je oboznámiť sa s významom výnimiek a vyskúšať si ich použitie. Tiež sa naučíme vytvárať unit testy pre naše aplikácie.

Postup

Krok č. 44

Použite výnimky pri spracovaní vstupov.

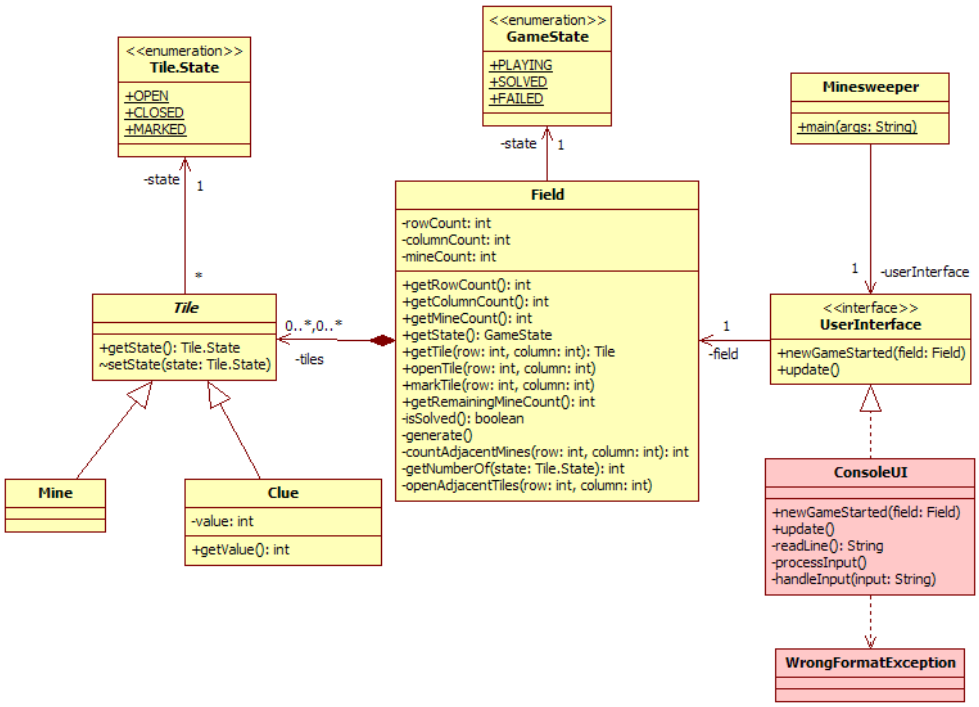
Úloha 0.34: Pridajte do projektu triedu `WrongFormatException` do balíka `minesweeper.consoleui` definujúcu výnimky označujúce zlý formát vstupu. Zdrojový kód danej triedy je WrongFormatException.java.

Úloha 0.35: V triede `ConsoleUI` implementujte súkromnú metódu `void handleInput(String input) throws WrongFormatException` , ktorá zabezpečí spracovanie vstupného reťazca. Presuňte do tejto metódy existujúce spracovanie vstupného reťazca z metódy `void processInput()`. V prípade zle zadaného formátu táto metóda vyhodí výnimku `WrongFormatException`.

Úloha 0.36: Do metódy `void processInput()` vložte volanie metódy `void handleInput(String input) throws WrongFormatException`. Použitie príkaz `try { ... } catch (WrongFormatException ex) {...}` na odchytenie výnimky a jej spracovanie (vypísanie `ex.getMessage()`).

Krok č. 45

Aktuálny model tried aplikácie je nasledujúcom obrázku.



Obrázok 11: Diagram tried aplikácie **Minesweeper** (na konci cvičenia 7)

Krok č. 46

V hre **Minesweeper** je zvyčajne vypísaný počet neoznačených mín, ktorý je definovaný ako počet všetkých mín v hracom poli mínus počet značiek na označenie míny. Nasledujúcim krokom je preto dodanie tejto podpory do hry.

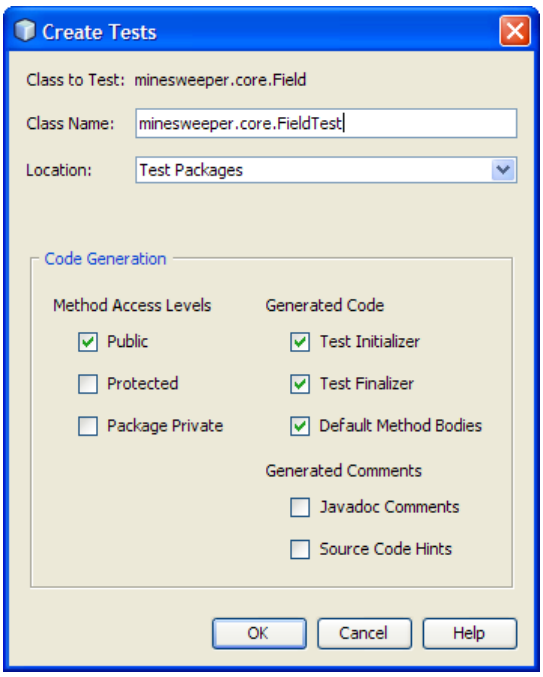
Úloha 0.37: Doplňte funkčnosť do metódy `update()` v triede `ConsoleUI` , ktorá zabezpečí výpis s uvedením počtu neoznačených mín. Pre získanie počtu neoznačených mín implementujte verejnú metódu `int getRemainingMineCount()` v triede `Field`.

Poznámka: V implementovanej metóde `int getRemainingMineCount()` použite metódu `int getNumberOf(Tile.State state)` triedy `Field` realizovanú v cvičení 5, ktorá vráti počet dlaždíc v požadovanom stave.

Krok č. 47

Pri realizácii programových systémov vzniká požiadavka testovania funkčných častí ako aj testovania integrácie týchto častí. Pre testovanie programov v prostredí NetBeans je možné použiť nástroj s podporou automatického generovania JUnit testov a ich spúšťania. Cieľom je vytvoriť testy pre testovanie generovania hracieho poľa a overenie implementácie zmien stavov hracieho poľa. Použitie nástroja pre automatické generovanie prototypu testu pre členské metódy vybranej triedy v prostredí NetBeans IDE a ich spustenie je nasledovné.

- Prostredníctvom kontextového menu nad názvom súboru `Field.java` v záložke „Projects" zvoľte „Tools > Create JUnit Tests".
- V dialógu „Select JUnit Version" zvoľte „JUnit 4.x".
- Zobrazí sa nasledujúce okno s možnosťou výberu metód na základe ich úrovne viditeľnosti, voliteľného vygenerovania kódu s možnosťou vygenerovania dokumentačných komentárov.



- Na základe vami zvolených možností sa vygeneruje testovací súbor s množinou testovacích metód. V prípade, že chcete testovať komplexnejšiu časť ako len jednu metódu triedy Field, jednoducho zmažete nevyhovujúce vygenerované testovacie metódy, resp. upravte názvy testovacích metód. Vytvorený testovací súbor bude umiestnený v časti „Test Packages" viditeľný v záložke „Projects" pre projekt `Minesweeper`.
- Pre spustenie testu sú možné nasledujúce postupy:
  - V kontextovom menu nad záložkou projektu zvoliť možnosť „Test Project".
  - V hlavnom menu zvoliť „Run > Test "Minesweeper"".

Úloha 0.38: Vygenerujte pre triedu `Field` testovací súbor `FieldTest`.

Vo vytvorenej triede FieldTest definujte nasledujúce súkromné konštanty:

- Počet riadkov hracieho poľa: `static final int ROWS = 9;`
- Počet stĺpcov hracieho poľa: `static final int COLUMNS = 9;`
- Počet mín v hracom poli: `static final int MINES = 10;`

Krok č. 48

Skopírujte nižšie uvedenú implementáciu metódy `void isSolved()` do triedy `FieldTest` projektu **Minesweeper**.

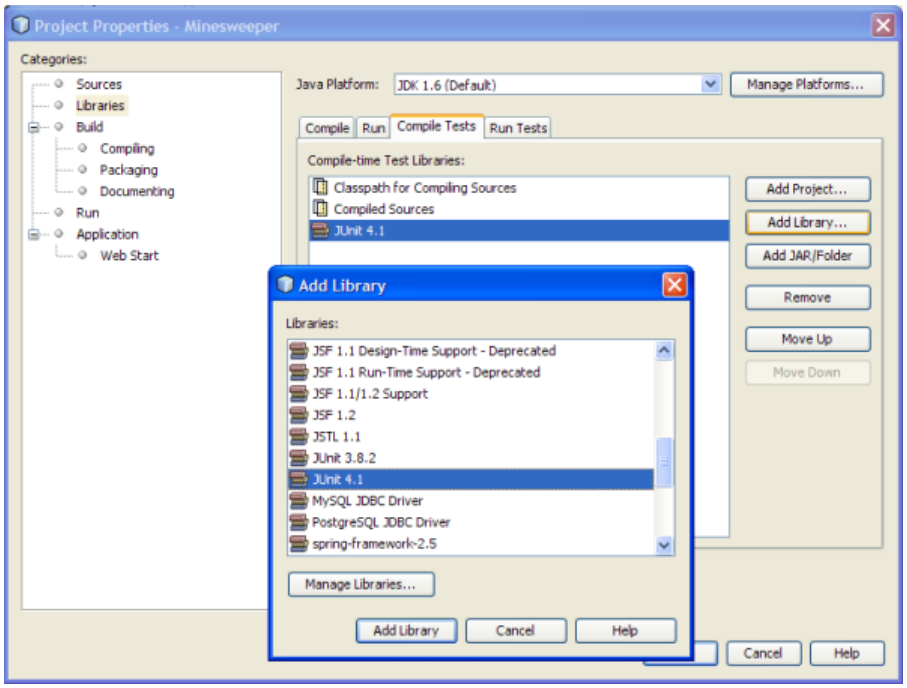
```
@Test
public void isSolved() {
    Field field = new Field(ROWS, COLUMNS, MINES);

    assertEquals(GameState.PLAYING, field.getState());

    int open = 0;
    for(int row = 0; row < field.getRowCount(); row++) {
        for(int column = 0; column < field.getColumnCount(); column++) {
            if(field.getTile(row, column) instanceof Clue) {
                field.openTile(row, column);
                open++;
            }
            if(field.getRowCount() * field.getColumnCount() - open == field.getMineCount()) {
                assertEquals(GameState.SOLVED, field.getState());
```

```
16              } else {
17                  assertNotSame(GameState.FAILED, field.getState());
18              }
19          }
20      }
21
22      assertEquals(GameState.SOLVED, field.getState());
23  }
```

Poznámka: Uistite sa, že v časti importovania zdrojov je uvedený statický import `import static org.junit.Assert.*;` ako aj import pre anotáciu `@Test` uvedený nasledovne `import org.junit.Test;`.

Poznámka: Nezabudnite, že každej testovacej metóde musí predchádzať anotácia `@Test`.

Poznámka: V pripade, že ste predtým nezvolili JUnit 4.x je možné ho pridať aj dodatočne pre projekt. Zvoľte nad projektom v záložke „Projects" v kontextovom menu voľbu „Properties".



Táto metóda testuje implementáciu rozhodnutia či je hra úspešne vyriešená. V prvom kroku je potrebné vytvoriť objekt `field` pre hracie pole typu `Field`, čím sa inicializuje aj počiatočný stav hry. Nad týmto hracím poľom testujeme nasledujúce požiadavky:

- Po vytvorení hracieho poľa musí byť hracie pole v stave priebehu hry - `GameState.PLAYING`. Pre tento typ testu použijeme výraz `assertEquals(GameState.PLAYING, field.getState())`.
- Po odkrytí dlaždice typu `Clue` je potrebné testovať stav úspešného ukončenia hry. Ak platí **počet všetkých dlaždíc - počet odokrytých dlaždíc = počet mín** potom musí byť hra v stave úspešného ukončenia. Pre test úspešného ukončenia hry použijeme výraz `assertEquals(GameState.SOLVED, field.getState())`.
- Po odkrytí dlaždice typu `Clue` je potrebné testovať aby nenastal stav neúspešného ukončenia hry. Pre tento typ testu použijeme výraz `assertNotSame(GameState.FAILED, field.getState())`.
- Po odkrytí všetkých dlaždíc typu `Clue` musí byť hra v stave úspešného ukončenia hry. Pre tento typ testu použijeme už vyššie uvedený výraz `assertEquals(GameState.SOLVED, field.getState())`.

Úloha 0.39: Spustite test pre projekt **Minesweeper**.

Krok č. 49

Ďalším z možných testov je testovanie správnosti vygenerovania hracieho poľa (trieda `Field`). Je potrebné otestovať nasledujúce požiadavky na vygenerované hracie pole:

- Počet riadkov, stĺpcov a mín v hracom poli zodpovedá zadaným počtom `assertEquals(ROWS, field.getRowCount())`, `assertEquals(COLUMNS, field.getColumnCount())`, `assertEquals(MINES, field.getMineCount())`.
- Každá položka hracieho poľa musí byť reprezentovaná dlaždicou a nesmie byť rovná hodnote `null`. Pre každú položku hracieho poľa je teda potrebné testovať výraz `assertNotNull(field.getTile(row, column))`, kde premenná `field` je objektom hracieho poľa typu `Field`.
- Počet vygenerovaných dlaždíc typu `Mine` musí byť zhodný s požadovaným počtom mín. Pre test je potrebné získať počet dlaždíc typu `Mine` v hracom poli (napríklad do premennej `mineCount`) a aplikovať test `assertEquals(MINES, mineCount)`.
- Počet vygenerovaných dlaždíc typu `Clue` musí byť zhodný s **počtom všetkých dlaždíc - počet mín**. Pre test je potrebné získať počet dlaždíc typu `Clue` v hracom poli (napríklad do premennej `clueCount`) a aplikovať test `assertEquals(ROWS * COLUMNS - MINES, clueCount)`.

Úloha 0.40: Pridajte verejnú metódu `void generate()` do testovacej triedy `FieldTest`. Implementujte ju podľa požiadaviek uvedených v tomto bode. Overte týmto testom správnosť vygenerovania hracieho poľa.

Krok č. 50

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Zdroje

[1] Zdrojové kódy:
- WrongFormatException.java

Kolekcie (Minesweeper Task 6)

Ciele

1. Oboznámiť sa s významom a druhmi kolekcií v jazyku Java. (krok 1, 2)
2. Naučiť sa používať implementáciu `ArrayList` rozhrania `List`. (krok 2, 3)
3. Naučiť sa používať všeobecné algoritmy pre prácu s kolekciami v triede `Collections`.
4. Pochopiť sekvenčný prístup k elementom kolekcie prostredníctvom objektu `Iterator`.
5. Implementovať meranie trvania hry. (krok 4)

Úvod

Ďalší krok pri implementácii hry **Minesweeper** je pridanie podpory pre meranie trvania hry. Popri tom sa naučíme používať kolekcie a ukážeme si ako implementovať návhový vzor **Singleton**.

Postup

Krok č. 51

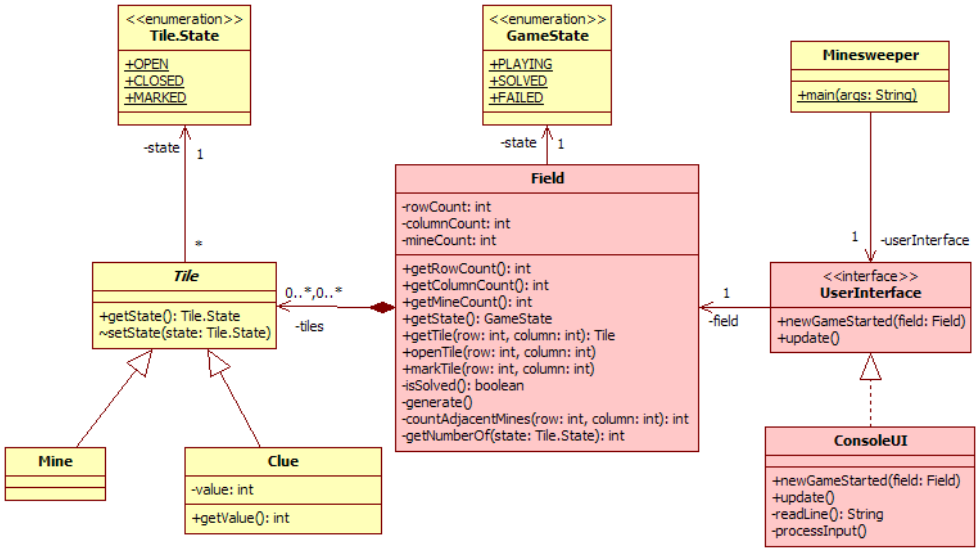
Úloha 0.41: Pridajte do triedy `Minesweeper` súkromnú členskú premennú `long startMillis`. Táto premenná bude slúžiť na uloženie času začiatku hrania hry. Nastavte danú premennú na začiatku hrania hry použitím metódy `System.currentTimeMillis().`

Úloha 0.42: Pridajte do triedy `Minesweeper` verejnú metódu `int getPlayingSeconds()`, ktorej návratovou hodnotou je počet sekúnd hrania hry.

Poznámka: Na určenie počtu sekúnd hrania hry použite aktuálny čas (`System.currentTimeMillis()`) a hodnotu premennej predstavujúcu čas začatia hrania hry`startMillis`. Nezabudnite na to, že dané hodnoty sú v milisekundách.

Krok č. 52

Rozšírte aplikáciu o možnosť ukladania času najlepších hráčov. Postupujte v súlade s uvedeným modelom.



Obrázok 12: Diagram tried aplikácie **Minesweeper** (cvičenie 8)

Úloha 0.43: Pridajte do projektu do balíka minesweeper zdrojový text BestTimes.java.

Úloha 0.44: Oboznámte sa s obsahom daného zdrojového textu. Daný zdrojový text obsahuje triedu `BestTimes` a statickú vnorenú triedu `PlayerTime`. Trieda `BestTimes` definuje manipuláciu s hráčmi a ich dosiahnutými časmi v hre. Trieda `PlayerTime` definuje objekty slúžiace na uloženie údajov o hráčovi (meno) a jeho čase.

Úloha 0.45: V triede `PlayerTime` implementujte verejné metódy `String getName()` a `int getTime()`.

Poznámka: Použite refaktorizáciu `Encapsulate Fields` prezentovanú na 4. cvičení.

Úloha 0.46: Implementujte verejnú metódu `String toString()` v triede `PlayerTime`, ktorá vráti textovú reprezentáciu času hráča (meno a čas).

Úloha 0.47: V triede `PlayerTime` implementujte rozhranie `Comparable` pridaním konštrukcie `implements Comparable<PlayerTime>` a implementujte metódu `int compareTo(PlayerTime o)`. Objekty triedy `PlayerTime` majú byť usporiadané podľa času hrania (`time`).

Poznámka: Rozhranie `Comparable` je definované v balíku `java.lang` a slúži na definovanie usporiadania objektov. Preštudujte dokumentáciu k danému rozhraniu. Vďaka implementácii rozhrania bude definované usporiadanie časov hráčov.

Úloha 0.48: Implementujte verejnú metódu `public void addPlayerTime(String name, int time)` v triede `BestTimes` , ktorá umožní pridanie mena hráča a času do zoznamu.

Poznámka: Zabezpečte aby objekty boli usporiadané v zozname. Použite statickú metódu `sort(List l)` triedy `Collections`.

Úloha 0.49: Implementujte verejnú metódu `String toString()` v triede `BestTimes`, ktorá vráti textovú reprezentáciu časov dosiahnutých hráčmi vrátane ich mien a poradia.

Poznámka: Pre vytváranie reťazca použite objekt triedy `Formatter`.

```
1 Formatter f = new Formatter();
2 //Do the formatting, see docs ...
3 return f.toString();
```

Krok č. 53

Ďalším krokom je rozšírenie triedy `Minesweeper` o používanie práve definovanej podpory pre uloženie časov hráčov.

Úloha 0.50: Pridajte do triedy `Minesweeper` súkromnú členskú premennú `BestTimes bestTimes`. Inicializujte túto premennú inštanciou triedy `BestTimes`.

Úloha 0.51: V triede `Minesweeper` implementujte verejnú metódu `BestTimes getBestTimes()`.

Poznámka: Použite refaktorizáciu `Encapsulate Field` s prezentovanú na 4. cvičení.

Krok č. 54

Upravte implementáciu triedy `Minesweeper` aby zodpovedala návrhovému vzoru **Singleton**.

Úloha 0.52: Pridajte do triedy `Minesweeper` súkromnú **statickú** premennú `Minesweeper instance`. Inicializujte túto premennú na prvom riadku konštruktora nasledovne: `instance = this`.

Úloha 0.53: V triede `Minesweeper` implementujte verejnú **statickú** metódu `Minesweeper getInstance()`, ktorej návratová hodnota je jediná inštancia triedy `Minesweeper`.

Krok č. 55

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Zdroje

[1] Zdrojové kódy:
- BestTimes.java

Doplňujúce úlohy

Úloha 0.54: Doplňte do triedy `ConsoleUI` do metódu `void update()` výpis času hrania hry.

Úloha 0.55: Doplňte do triedy `BestTimes` metódu void `reset()` slúžiacu na zrušenie záznamov o hráčoch a ich časoch.

Údajové prúdy (Minesweeper Task 7)

Ciele

1. Oboznámiť sa s významom a použitím prúdov údajov v jazyku Java. (krok 1, 2, 4, 5)
2. Oboznámiť sa s významom a použitím serializácie v jazyku Java. (krok 2, 3)
3. Implementovať podporu pre výber a uloženie nastavení v hre Minesweeper.

Úvod

Na dnešnom cvičení si vyskúšame prácu s údajovými prúdmi a do hry pridáme možnosť nastavenia obtiažnosti hry.

Postup

Krok č. 56

Nasledujúcim krokom v rámci implementácie hry **Minesweeper** je pridanie podpory pre uloženie nastavení hry. V nastaveniach bude uchovávaná informácia o naposledy zvolenej obtiažnosti hry.

Úloha 0.56: Vytvorte triedu `Settings`, ktorá definuje nastavenie hry a zabezpečuje uloženie do súboru a získanie nastavenia zo súboru. Túto triedu umiestnite do balíka `minesweeper`. Nastavenie hry bude definovať veľkosť hracieho poľa a počet mín.

Úloha 0.57: Vzhľadom na zámer serializácie nastavenia pridajte v definícii triedy `Settings` implementáciu rozhrania`Serializable`, ktoré sa nachádza v balíku `java.io`.

Poznámka: Rozhranie `Serializable` nedefinuje žiadnu metódu. Slúži len na označenie možnosti objektov tejto triedy serializovať ich (návrhový vzor **Marker Interface**).

Úloha 0.58: Vo vytvorenej triede `Settings` vytvorte členské premenné `rowCount`, `columnCount` a `mineCount` typu `private final int`, ktoré budú definovať veľkosť hracieho poľa a počet mín. Vytvorte v tejto triede konštruktor `public Settings(int rowCount, int columnCount, int mineCount)`. V tele konštruktora nastavte hodnoty členských premenných s rovnakými menami (použite kľúčové slovo `this`).

Úloha 0.59: Použitím nástroja pre refaktorizáciu zdrojového kódu vygenerujte pre vytvorené premenné metódy na získanie ich hodnôt:

- `public int getRowCount()`
- `public int getColumnCount()`
- `public int getMineCount()`

Úloha 0.60: Pridajte do triedy `Settings` tri statické premenné definujúce rozmery a počet mín hracieho poľa pre začiatočníkov (`BEGINNER`), mierne pokročilých (`INTERMEDIATE`) a pokročilých (`EXPERT`).

Tieto premenné sú typu `public static final Settings`. Veľkosti hracieho poľa a počty mín sú definované nasledovne:

- Hracie pole pre začiatočníkov - veľkosť 9x9, počet mín 10 - `BEGINNER = new Settings(9, 9, 10)`.
- Hracie pole pre mierne pokročilých - veľkosť 16x16, počet mín 40 - `INTERMEDIATE = new Settings(16, 16, 40)`.
- Hracie pole pre pokročilých - veľkosť 16x30, počet mín 99 - `EXPERT = new Settings(16, 30, 99)`.

Úloha 0.61: Definujte v triede `Settings` premennú `SETTING_FILE` typu `private static final String` obsahujúcu informáciu o súbore, kde bude nastavenie hry uložené, resp. z ktorého bude nastavenie hry získané pri jej spustení.

Poznámka: Túto premennú môžete nastaviť nasledovne `SETTING_FILE = System.getProperty("user.home") + System.getProperty("file.separator") + "minesweeper.settings";`.

Úloha 0.62: Pre získanie informácie o nastavenej obtiažnosti hry je potrebné v triede `Settings` prekryť metódu public boolean equals(Object o) triedy `Object`. Táto metóda sa používa na porovnávanie dvoch objektov.

V našom prípade bude neskôr použitá pri označení zvolenej obtiažnosti hry v grafickom používateľskom rozhraní, kde bude potrebné vykonať porovnanie zvolenej obtiažnosti hry s preddefinovanými, ktoré sú reprezentované premennými `BEGINNER`, `INTERMEDIATE` a `EXPERT` uvedenými vyššie. Podmienkou, kedy dva objekty typu `Settings` sú rovnaké je platnosť nasledujúcich podmienok:

- Majú rovnaký počet riadkov.
- Majú rovnaký počet stĺpcov.
- Majú rovnaký počet mín.

Poznámka: Pre porovnávanie objektov nie je vhodné používať operátor `==` v prípade, ak je potrebné vykonať porovnanie objektov na základe ich stavu (hodnôt premenných). Operátor `==` pri porovnaní hodnôt referečných typov porovnáva adresy umiestnenia objektov v pamäti!

Úloha 0.63: Prekryte v triede `Settings` vykonávanie metódy public int hashCode() definovanej v triede `Object`.

Poznámka: Je to nutné z dôvodu nami definovanej implementácie metódy `public boolean equals(Object o)`. Objekty, pre ktoré platí rovnosť na základe porovnania metódou `equals`, majú mať rovnakú návratovú hodnotu pri použití metódy `hashCode`. Návratovou hodnotou implementácie tejto metódy bude hodnota výrazu **počet riadkov * počet stĺpcov * počet mín**.

Krok č. 57

V ďalšom kroku je potrebné realizovať uloženie nastavenia hry tak, aby nastavenie hry mohlo byť inicializované pri jej spustení. Nastavenia hry budú ukladané do súboru prostredníctvom serializácie objektov a pri spustení hry bude nastavenie načítané zo súboru.

Úloha 0.64: Implementujte metódu `public void save()` v triede `Settings`, ktorá uloží nastavenie zvolenej obtiažnosti hry do súboru použitím prúdu `FileOutputStream` a prúdu `ObjectOutputStream` pre serializáciu objektov. Pre identifikáciu súboru, do ktorého bude nastavenie uložené použite hodnotu premennej `SETTING_FILE`.

Poznámka: Metóda `public void save()` je metódou objektu a teda jej úlohou je vykonať uloženie tohto objektu (objektu nad ktorým je táto metóda aplikovaná). Preto je potrebné použiť kľúčové slovo `this`.

Poznámka: Odchytenie výnimky pri práci so serializáciou objektu (ale aj v ostatných prípadoch vyžadujúcich odchytenie výnimky) je možné realizovať v prostredí NetBeans nasledovne:

- Označte kód, pre ktorý je požadované spracovanie výnimky.
- Nad ľavým okrajom sa objaví žiarovka nad ktorou zvoľte „Surround with try ...".

Úloha 0.65: Implementujte statickú verejnú metódu `public static Settings load()` v triede `Settings`, ktorá načíta uložené nastavenie obtiažnosti hry zo súboru použitím prúdov `ObjectInputStream` a `FileInputStream`. Pre identifikáciu súboru, z ktorého bude serializovaný objekt získaný použite hodnotu premennej `SETTING_FILE`. V prípade, že sa nepodarí deserializovať objekt typu `Settings` s uloženým nastavením používateľa, bude návratovou hodnotou objekt s obtiažnosťou nastavenou v premennej `BEGINNER`.

Krok č. 58

Posledným krokom pri implementácii ukladania/obnovenia nastavenia obtiažnosti hry **Minesweeper** je vykonanie úpravy v triede Minesweeper, ktorá sa nachádza v balíku `minesweeper`.

Úloha 0.66: Definujte v triede `Minesweeper` súkromnú premennú setting typu `Settings`.

Úloha 0.67: Vygenerujte v triede `Minesweeper` metódy `public Settings getSetting()` a `public void setSetting(Settings setting)`.

Úloha 0.68: Pri spustení hry **Minesweeper** inicializujte premennú `setting` posledným uloženým nastavením obtiažnosti hry. Na tento účel použite statickú metódu `Settings load()` triedy `Settings` v konštruktore triedy `Minesweeper`.

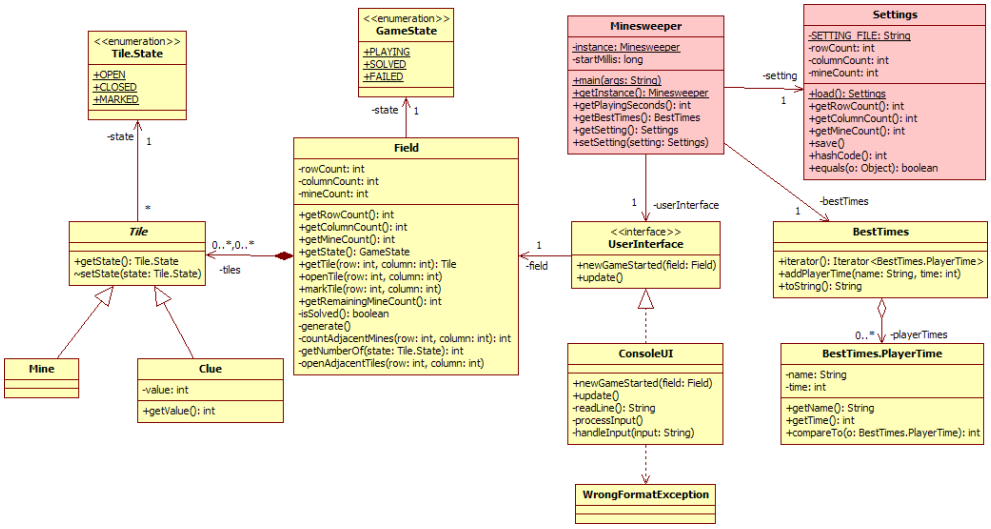
Úloha 0.69: Pri zmene nastavení v metóde `public void setSetting(Settings setting)` zabezpečte uloženie nových nastavení pomocou metódy `void save()` nad objektom `setting`.

Krok č. 59

Na otestovanie funkčnosti implementovanej triedy Settings vytvorte vhodný JUnit test.

Krok č. 60

Aktuálny model tried aplikácie je nasledujúcom obrázku.



Obrázok 13: Diagram tried aplikácie Minesweeper (na konci cvičenia 7)

Krok č. 61

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Zdroje

[1] Metóda public boolean equals(Object o) v Java API.
[2] Metóda public int hashCode() v Java API.

Grafické používateľské rozhranie (Minesweeper Task 8)

Ciele

1. Oboznámiť sa s tvorbou grafického používateľského rozhrania. (krok 1, 2, 4, 5, 6)
2. Oboznámiť sa s udalosťami riadeným vykonávaním programu. (krok 2, 3)
3. Implementovať grafické používateľské rozhranie Swing pre hru Minesweeper.

Úvod

Dnes vytvoríme grafické používateľské rozhranie pre našu hru **Minesweeper**.

Postup

Krok č. 62

Pridajte do triedy `Minesweeper` nasledujúcu metódu. Táto metóda sa bude používať pre začatie hrania novej hry.

```
public void newGame() {
    Field field = new Field(setting.getRowCount(), setting.getColumnCount(), setting.getMineCount());
    startMillis = System.currentTimeMillis();
    userInterface.newGameStarted(field);
}
```

Teraz ju možete použiť v konštruktore `Minesweeper` na začatie prvej hry.

Krok č. 63

V nasledujúcom kroku pridajte do projektu **Minesweeper** triedy pre grafické používateľské rozhranie.

Úloha 0.70: Pridajte do projektu (do adresára `src`) obsah archívu swingui.zip.

Archív obsahuje adresár `img` s obrázkami a balík `minesweeper.swingui`. V balíku `minesweeper.swingui` sú triedy `TileComponent` a `SwingUI`. Oboznámte sa s triedami `TileComponent` a `SwingUI`.

- Trieda `TileComponent` je grafický komponent slúžiaci pre vykreslenie jednej dlaždice hracieho poľa.
- Trieda `SwingUI` predstavuje okno hry a zároveň implementuje rozhranie `UserInterface`.

Krok č. 64

Nasledujúcim krokom je zobrazenie hracej plochy v okne aplikácie.

Úloha 0.71: Implementujte metódu `public void newGameStarted(Field field)` triedy `SwingUI`.

V rámci tejto metódy je potrebné vykonať nasledujúce kroky:

- Inicializovať premennú objektu `field` argumentom metódy `newGameStarted(Field field)`.
- Odstrániť všetky komponenty, ktoré obsahuje panel `contentPanel` prostredníctvom volania `contentPanel.removeAll()` (Komponenty sa odstraňujú z dôvodu vytvorenia nového hracieho poľa). Nastavte pre panel `contentPanel` manažér rozmiestnenia na typ `GridLayout` prostredníctvom aplikácie

```
1 contentPanel.setLayout(new GridLayout(field.getRowCount(), field.getColumnCount()))
```

- Vytvoriť pre každú dlaždicu hracieho poľa grafický komponent triedy `TileComponent` spätý s danou dlaždicou.
- Tento komponent následne pridajte do kontajnera hracieho poľa `contentPanel` pomocou metódy `Component add(Component comp)`.
- Pre každý komponent dlaždice je potrebné nastaviť spracovanie udalosti pre stlačenie tlačidla myši - `MouseListener`. Pre nastavenie spracovania udalosti použite metódu `void addMouseListener(MouseListener l)`. Trieda `SwingUI` implementuje rozhranie `MouseListener`.

```
1 public class SwingUI extends JFrame implements UserInterface, MouseListener {...}
```

- Aplikovať metódu `update()` triedy `SwingUI` pre zobrazenie počiatočného stavu hracieho poľa.
- Aplikovať metódu `pack()` triedy `SwingUI` zabezpečujúcu rozloženie komponentov hlavného okna hry **Minesweeper** s cieľom minimalizácie veľkosti okna (kompaktný tvar).

Úloha 0.72: Implementujte metódu `public void update()` triedy `SwingUI`. V tejto metóde je nutné nad každým grafickým komponentom triedy `TileComponent` aplikovať metódu `updateStyle()`. Taktiež je potrebné aktualizovať počítadlo zostávajúcich neoznačených mín aplikovaním metódy `setMinesLeftLabelText()` triedy `SwingUI`.

Poznámka: Komponenty `TileComponent` sú uložené priamo v paneli `contentPanel`. Prechádzať prvkami môžete prostredníctvom metód `contentPanel.getComponentCount()` a `contentPanel.getComponent(index)`.

Krok č. 65

Ďalším krokom je definovanie spracovania udalosti nad dlaždicami hracej plochy hry **Minesweeper**.

Úloha 0.73: Implementujte metódu `public void mousePressed(MouseEvent e)` triedy `SwingUI`. Táto metóda je implementáciou reakcie na stlačenie tlačidla myši nad grafickým komponentom triedy `TileComponent`.

Vykonanie tejto metódy je určené nasledujúcimi krokmi:

- Podmienkou reakcie na stlačenie myši je stav hry `GameState.PLAYING`. V inom prípade by nemalo byť hracie pole menené.
- Pri stlačení ľavého tlačidla myši je potrebné aplikovať metódu `void openTile(int row, int column)` objektu hracieho poľa triedy `Field`. Pozícia grafického komponentu dlaždice v hracom poli je prístupná v objekte grafického komponentu `TileComponent`.

Poznámka: Komponent dlaždice nad ktorou bola myš stlačená je možné získať z objektu udalosti nasledovne: `TileComponent button = (TileComponent)e.getSource();`

Poznámka: Pre identifikovanie stlačenia ľavého tlačidla myši použite statickú metódu `isLeftMouseButton(MouseEvent e)` triedy `SwingUtilities`.

- V prípade, že sa zmenil stav hry na stav `GameState.FAILED`, je potrebné vypísať používateľovi v dialógovom okne informáciu o chybnom ukončení hry.

Poznámka: Pre zobrazenie dialógového okna s požadovanou informáciou použite statickú metódu `void showMessageDialog(Component parentComponent, Object message, String title, int messageType)` triedy `JOptionPane`.

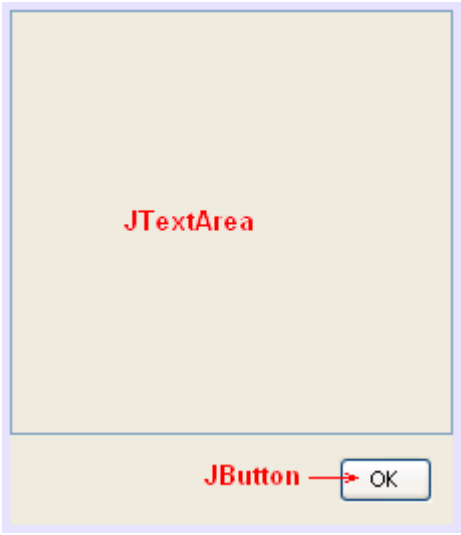
- V prípade, že sa zmenil stav hry na stav `GameState.SOLVED`, je potrebné vypísať používateľovi v dialógovom okne informáciu o úspešnom ukončení hry. Taktiež je potrebné zapísať používateľa do tabuľky najlepších časov použitím metódy `void addPlayerTime(String name, int time)` triedy `BestTimes`. Pre získanie času hrania hry použite metódu `getPlayingSeconds()` objektu triedy `Minesweeper`. Pre získanie mena používateľa použite výraz `System.getProperty("user.name")`, ktorý poskytne hodnotu systémovej premennej prostredia `user.name` predstavujúcu meno používateľa prihláseného do systému.
- Pri stlačení pravého tlačidla myši je potrebné aplikovať metódu void `markTile(int row, int column)` objektu hracieho poľa triedy `Field`. `Pre identifikovanie stlačenia pravého tlačidla myši použite statickú metódu isRightMouseButton(MouseEvent e) triedy SwingUtilities.`
- Po vykonaní odkrytia resp. označenia dlaždice je potrebné prekresliť hracie pole aplikovaním metódy `update()` triedy `SwingUI`.

Krok č. 66

V tomto kroku je potrebné realizovať zobrazenie zoznamu najlepších časov hráčov hry **Minesweeper**.

Úloha 0.74: Vytvorte dialóg na zobrazenie času najlepších hráčov.

Inšpirujte sa nasledujúcim vzorom:



Obrázok 14: Vzor dialógu pre zobrazenie času najlepších hráčov

- Vytvorte triedu `BestTimesDialog` typu `JDialog` („File > New File... > Java GUI Forms > Swing GUI Forms"). Nezabudnite zaradiť túto triedu do balíka`minesweeper.swingui`.
- Odstráňte metódu `main` v triede`BestTimesDialog`.
- V konštruktore triedy `BestTimesDialog` implementujte zobrazenie najlepšieho skóre získaného aplikovaním metódy `getBestTimes()` nad objektom triedy `Minesweeper`. Informácie o časoch zobrazte do grafického prvku `JTextArea` (pomocou metódy `void setText(String t)`).

  Poznámka: Pre vycentrovanie dialógového okna nad oknom hracej plochy použite metódu `setLocationRelativeTo(Component c)` nad objektom dialógového okna.

- Zaregistrujte nad tlačidlom s textom „OK" obsluhu udalosti `ActionListener`. V prostredí NetBeans v zobrazení návrhu dialógového okna označte komponent tlačidla. V okne s názvom „Properties" zvolíte záložku „Events" a kurzor nastavíte do textového poľa pre zadanie názvu udalosti „actionPerformed". Stlačte „Enter". V zdrojovom kóde sa automaticky vytvorí metóda pre spracovanie udalosti. Ak bude grafický komponent tlačidla pomenovaný názvom `okButton`, tak metóda bude vygenerovaná v nasledujúcej forme:

```
1 private void okButtonActionPerformed(java.awt.event.ActionEvent evt) {
2     // TODO add your handling code here:
3 }
```

  Implementujte spracovanie udalosti pre stlačenie tlačidla nasledovne:

```
1 setVisible(false);
2 dispose();
```

  Daný kód zatvorí dialógové okno a uvoľní zdroje späté s oknom.
- V triede `SwingUI` dodajte implementáciu stlačenia menu „Game > Best Times...". Na vyhľadanie položky menu `bestTimesMenuItem` použite okno „Inspector". Postupujte podľa postupu uvedenom hore. Implementácia reakcie na udalosť je nasledovná.

```
1 new BestTimesDialog(this, true).setVisible(true);
```

Krok č. 67

Pre zaradenie implementovaného grafického používateľského rozhrania do hry **Minesweeper** je potrebné vykonať úpravu v hlavnej triede aplikácie podľa nasledujúcich úloh.

Úloha 0.75: V triede `Minesweeper` definujte premennú `private static final String DEFAULT_UI = "swing"`. Na základe hodnoty tejto premennej sa určí, aký typ používateľského rozhrania bude použitý (`SwingUI`, alebo `ConsoleUI`).

Úloha 0.76: Implementujte metódu `private UserInterface create(String name)` v triede `Minesweeper`.

Metóda na základe mena grafického rozhrania vytvorí požadovaný objekt rozhrania (návrhový vzor **Factory Method**).

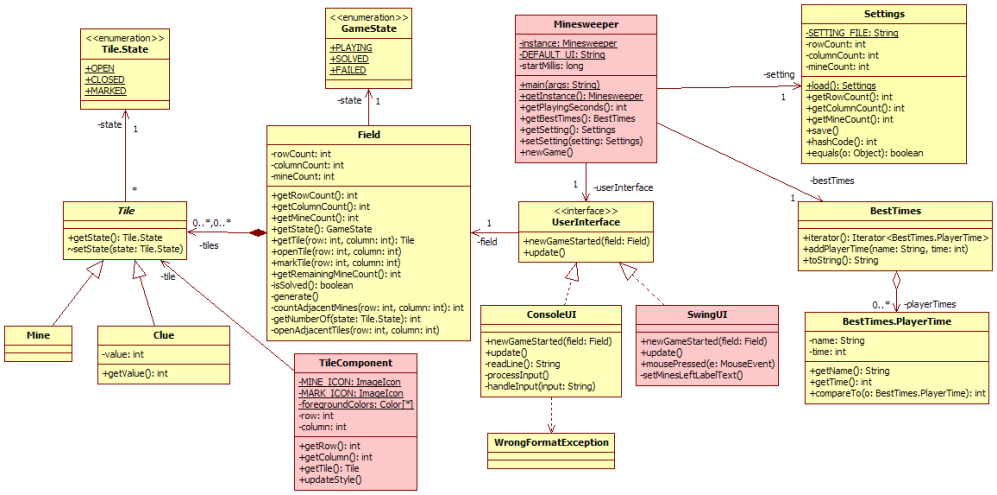
- Ak hodnota parametra name je rovná "`swing`", tak návratovou hodnotou bude objekt grafického používateľského rozhrania typu `SwingUI`.

- Ak hodnota parametra name je rovná hodnote ”`console`”, tak návratovou hodnotou bude objekt používateľského rozhrania typu `ConsoleUI`.
- Ak hodnota `DEFAULT_UI` nie je rovná ani jednej z vyššie uvedených hodnôt, tak vznikne objekt výnimky typu `RuntimeException` so správou ”No valid UI specified”.

Úloha 0.77: Pridajte aplikáciu metódy `create(DEFAULT_UI)` do konštruktora triedy `Minesweeper`.

Krok č. 68

Aktuálny model tried aplikácie je nasledujúcom obrázku.



Obrázok 15: Diagram tried aplikácie **Minesweeper** (na konci cvičenia 10)

Krok č. 69

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Zdroje

[1] Zdrojové kódy:
- swingui.zip

Doplňujúce úlohy

Úloha 0.78: Implementujte voľbu typu používateľského rozhrania pomocou argumentov príkazového riadku (prepínač `-swing` resp. `-console`).

Úloha 0.79: Dodajte podporu pre odkrývanie susedných dlaždíc pri stlačení oboch tlačidiel myši naraz.

Úloha 0.80: Rozšírte stav dlaždice o označenie otáznikom.

Vlákna (Minesweeper Task 9)

Ciele

1. Oboznámiť sa s významom a použitím vlákien v jazyku Java. (krok 1)
2. Implementovať časovač v hre Minesweeper.

Úvod

Úlohou dnešného cvičenia je pridanie podpory pre zobrazenie času hrania hry do hry **Minesweeper**.

Postup

Krok č. 70

Ďalší krok pri implementácii hry **Minesweeper** je pridanie podpory pre zobrazenie času hrania hry.

Úloha 0.81: Implementujte súkromnú metódu `void setTimeLabelText()` v triede `SwingUI`, ktorá sa nachádza v balíku `minesweeper.swingui`. Uvedená metóda nastaví text pre grafický komponent `timeLabel` typu `javax.swing.JLabel` tak, aby zobrazoval dĺžku trvania hry v okamihu zavolania metódy. Pre zistenie aktuálnej dĺžky trvania hry zavolajte metódu `getPlayingSeconds()` objektu triedy `Minesweeper` (pomocou statickej metódy `getInstance()`). Čas trvania hry sa uvádza v sekundách, pričom sa používajú vždy tri číslice na vyjadrenie počtu sekúnd hrania hry (napr. `008`).

Poznámka: Inšpirujte sa implementáciou metódy `void setMinesLeftLabelText()` v tej istej triede.

Úloha 0.82: Pridajte do triedy `SwingUI` súkromnú členskú premennú `timer` typu `javax.swing.Timer`. Časovač umožní v pravidelných intervaloch aktualizovať zobrazenie trvania hry.

Poznámka: Podrobné informácie o použití časovača nájdete v Java API.

Úloha 0.83: Do konštruktora triedy `SwingUI` doplňte vytvorenie objektu časovača.

Pre vytvorenie objektu triedy časovača `Timer` použite konštruktor `Timer(int delay, ActionListener listener)`. Parameter `delay` predstavuje trvanie časového intervalu v milisekundách, po ktorom nastane udalosť. Vzhľadom na meranie času po sekundách bude postačovať hodnota 100 (v milisekundách). Parameter `listener` predstavuje implementáciu obsluhy udalosti, ktorá nastane po uplynutí časového intervalu. Objekt implementujúci rozhranie `ActionListener` vytvorte ako anonymnú triedu podľa nasledujúceho zdrojového kódu:

```
ActionListener listener = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        //...Perform a task...
    }
};
```

Metóda `actionPerformed(ActionEvent evt)` má obsahovať aktualizáciu zobrazeného trvania hry (`void setTimeLabelText()`). Zobrazenie trvania hry je nutné aktualizovať iba vtedy, ak je hra v stave `GameState.PLAYING`, čo je možné zistiť použitím metódy `getState()` nad objektom hracieho poľa uloženého v členskej premennej `field`.

Úloha 0.84: V konštruktore triedy `SwingUI` spustite časovač zavolaním metódy `void start()` nad objektom `timer`.

Krok č. 71

Upozornenie: Vymažte projekt z disku a zrušte všetky vami vytvorené nastavenia v prostredí NetBeans!

Zdroje

[1] Trieda Timer v Java API.

Doplňujúce úlohy

Úloha 0.85: Modifikujte hru tak aby bolo možné zvoliť režim hrania hry oproti najlepšiemu času. Pridajte do menu zaškrtávaciu položku „Beat Best Time Mode". Ak je tento režim zvolený, zobrazený čas v sekundách ukazuje koľko času ešte ostáva do dokončenia hry v najlepšom čase.

Úloha 0.86: Dodajte do hry možnosť voľby veľkosti hracieho poľa a počtu mín. Navrhnite vhodný dialóg a pridajte položku do menu umožňujúcu zobrazenie dialógu.

JDBC (Minesweeper Task 10)

Ciele

1. Oboznámiť sa s tvorbou aplikácií komunikujúcich s databázovým systémom. (krok 1, 2)
2. Oboznámiť sa so základmi jazyka SQL. (krok 2, 3)
3. Implementovať funkčnosť pre prácu so zoznamom najlepších výsledkov uložených v databáze.

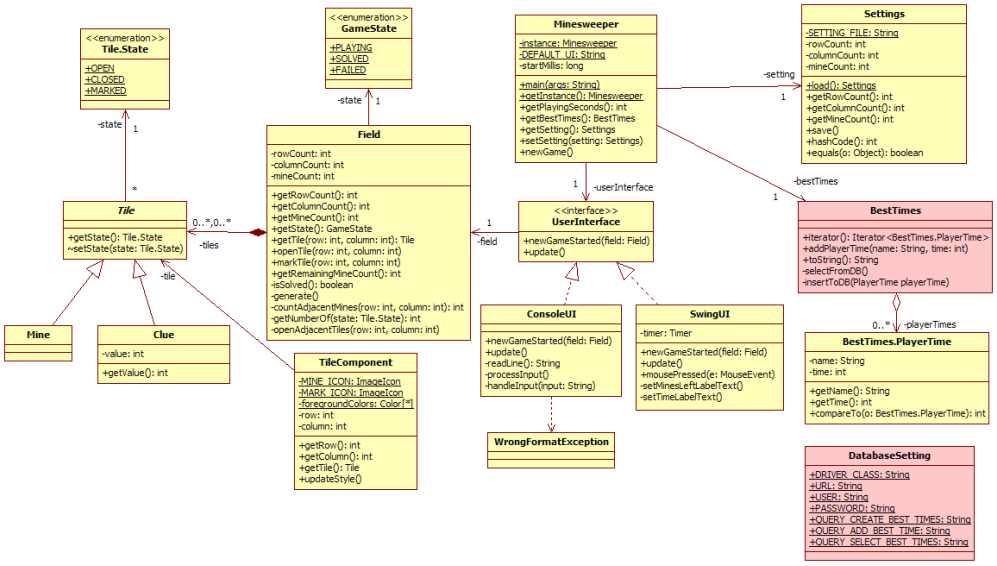
Úvod

Hlavným cieľom dnešného cvičenia je rozšírenie projektu o prácu s databázovým systémom. Pre naše potreby bol zvolený jednoduchý databázový systém Derby (Java DB), ktorý je implementovaný v jazyku Java. Napriek tomu nie je implementácia hry viazaná na tento systém a je možné použiť aj iný databázový systém (PostgreSQL, MySQL, HSQL a podobne).

Postup

Krok č. 72

Pre realizáciu cieľov dnešného cvičenia bude potrebné rozšíriť implementovaný systém podľa nasledujúceho modelu.



Obrázok 16: Diagram tried aplikácie **Minesweeper** (na konci cvičenia 12)

Úloha 0.87: Pridajte súbor DatabaseSetting.java do projektu hry **Minesweeper**. Uvedená trieda má definované konštanty obsahujúce konfiguračné údaje pre vytvorenie spojenia na databázový systém Derby ako aj údaje potrebné pre prácu s databázou (SQL príkazy).

Krok č. 73

V tomto kroku je potrebné rozšíriť triedu `BestTimes` tak, aby ukladala najlepšie skóre do databázy v prípade úspešného ukončenia hry a získavala toto skóre z databázy v prípade požiadavky na jeho zobrazenie.

Úloha 0.88: V triede `BestTimes` definujte súkromnú metódu `void insertToDB(PlayerTime playerTime)`, ktorá uloží objekt triedy `PlayerTime` do databázy.

Realizácia tejto metódy bude pozostávať z nasledujúcich krokov:

- Vytvorenie spojenia na databázový server so získaním objektu typu `Connection`. Trieda ovládača pre pripojenie na databázový server je určená hodnotou konštanty `DatabaseSetting.DRIVER_CLASS`. URL pre pripojenie na databázu je uvedené v hodnote konštanty `DatabaseSetting.URL`, meno a heslo v konštantách `DatabaseSetting.USER` a `DatabaseSetting.PASSWORD`.

```
Class.forName(DatabaseSetting.DRIVER_CLASS);
Connection connection = DriverManager.getConnection(DatabaseSetting.URL,
                                DatabaseSetting.USER, DatabaseSetting.PASSWORD);
```

- Vytvorenie databázovej tabuľky `player_time` v prípade, že ešte takáto tabuľka nie je vytvorená. Ak použijeme SQL príkaz `CREATE TABLE` nad tabuľkou, ktorá už existuje, vznikne výnimka. Za tohto predpokladu môže byť implementácia nasledovná:

```
Statement stm = connection.createStatement();
try {
    stm.executeUpdate(DatabaseSetting.QUERY_CREATE_BEST_TIMES);
} catch (Exception e) {
    //do not propagate exception, table may already exist
}
stm.close();
```

  Pre vytvorenie tabuľky je použitý nasledujúci príkaz jazyka SQL (Trieda `DatabaseSetting`, konštanta `QUERY_CREATE_BEST_TIMES`).

```
CREATE TABLE player_time (name VARCHAR(128) NOT NULL, best_time INT NOT NULL)
```

- Zápis mena hráča a jeho času do databázy. Pre realizáciu tejto úlohy je možné použiť objekt typu `PreparedStatement` nasledovne:

```
PreparedStatement pstm = connection.prepareStatement(DatabaseSetting.QUERY_ADD_BEST_TIME);
pstm.setString(1, playerTime.getName());
pstm.setInt(2, playerTime.getTime());
pstm.execute();
pstm.close();
```

  SQL príkaz pre uloženie mena a času hráča do tabuľky je nasledovný (trieda `DatabaseSetting`, konštanta `QUERY_ADD_BEST_TIME`):

```
INSERT INTO player_time (name, best_time) VALUES (?, ?)
```

- Uzatvorenie spojenia na databázu connection.close().

- Ošetrenie vzniku výnimky v prípade práce s databázovým serverom. Pri vzniku výnimky nech je na konzolu vypísaný text obsahujúci aj opis chyby poskytnutý ovládačom databázového systému. Text bude v tvare (premenná e je objektom výnimky):

```
"Exception occured during saving high score to database: " + e.getMessage()
```

Úloha 0.89: V triede `BestTimes` definujte súkromnú metódu `void selectFromDB()`, ktorá zabezpečí vytvorenie objektov typu `PlayerTime` s údajmi získanými z databázy a ich uloženie do zoznamu v členskej premennej `playerTimes`.

Realizácia metódy bude pozostávať z nasledujúcich krokov:

- Získanie objektu typu `Connection` , ktorý reprezentuje spojenie na databázový server.

```
Class.forName(DatabaseSetting.DRIVER_CLASS);
Connection connection = DriverManager.getConnection(DatabaseSetting.URL,
DatabaseSetting.USER, DatabaseSetting.PASSWORD);
```

- Vykonanie SQL príkazu `SELECT` nad databázovým serverom. Výsledkom je objekt typu `ResultSet` umožňujúci prechod v zozname získaných záznamov z databázy. Realizácia bude nasledujúca:

```
Statement stm = connection.createStatement();
ResultSet rs = stm.executeQuery(DatabaseSetting.QUERY_SELECT_BEST_TIMES);
```

SQL príkaz pre získanie mien a časov hráčov je nasledovný (trieda `DatabaseSetting`, konštanta `QUERY_SELECT_BEST_TIMES`):

```
SELECT name, best_time FROM player_time
```

- Vymazanie zoznamu objektov z členskej premennej `playerTimes` použitím metódy `clear()`, aby sme mohli aktualizovať tento zoznam aktuálnymi údajmi.
- Spracovanie získaných záznamov z databázového servera. Pre každý získaný záznam z databázy bude vytvorený objekt typu `PlayerTime` s nastavením príslušných údajov. Tento objekt bude pridaný do vytváraného zoznamu `playerTimes`.

```
while(rs.next()) {
    PlayerTime pt = new PlayerTime(rs.getString(1), rs.getInt(2));
    playerTimes.add(pt);
}
stm.close();
```

- Uzatvorenie spojenia na databázu `connection.close()`.
- Ošetrenie vzniku výnimky v prípade práce s databázovým serverom. Pri vzniku výnimky nech je na konzolu vypísaný text obsahujúci aj opis chyby poskytnutý ovládačom databázového servera. Text bude v tvare (premenná `e` je objektom výnimky):

```
"Exception occured during loading high score from database: " + e.getMessage()
```

- Zotrieďte údaje získané z databázy aby zodpovedali poradiu hráčov. Použite statickú metódu `sort(List list)` triedy `java.util.Collections`.

Úloha 0.90: Upravte metódu `void addPlayerTime(String name, int time)` tak, aby bol vytvorený objekt typu `PlayerTime` uložený do databázy použitím metódy `void insertToDB(PlayerTime playerTime)`.

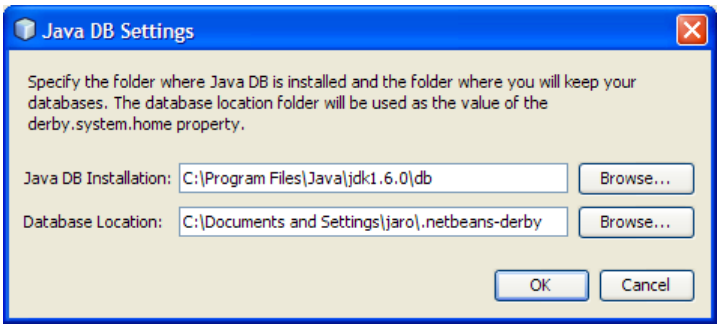
Úloha 0.91: Upravte metódu `String toString()` tak, aby sa aktualizoval zoznam časov hráčov `PlayerTime` v členskej premennej `playerTimes` podľa stavu v databáze použitím metódy `void selectFromDB()`.

Krok č. 74

Posledným krokom pre overenie realizovaných úloh je konfigurácia databázového servera Derby a nastavenie a vytvorenie databázy. Databázový systém Derby je súčasťou inštalácie JDK 6.

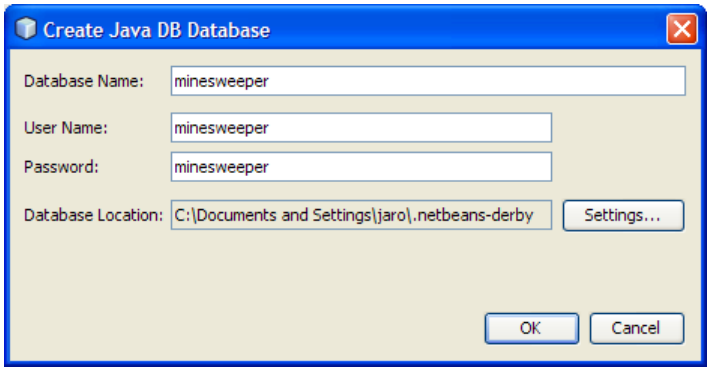
Úloha 0.92: Nakonfigurujte databázový systém Derby v prostredí Netbeans.

Zvoľte v menu „Window > Services". V strome služieb zvoľte uzol „Java DB". Otvorte kontextové menu a skontrolujte nastavenie pre DB (položka „Properties"). Nastavte adresár inštalácie Derby a adresár kde budú ukladané databázové súbory.



Úloha 0.93: Vytvorte databázu pre hru **Minesweeper**.

Vytvorte databázu zvolením položky „Create Database" v kontextovom menu. Napíšte názov databázy, meno používateľa, heslo používateľa.



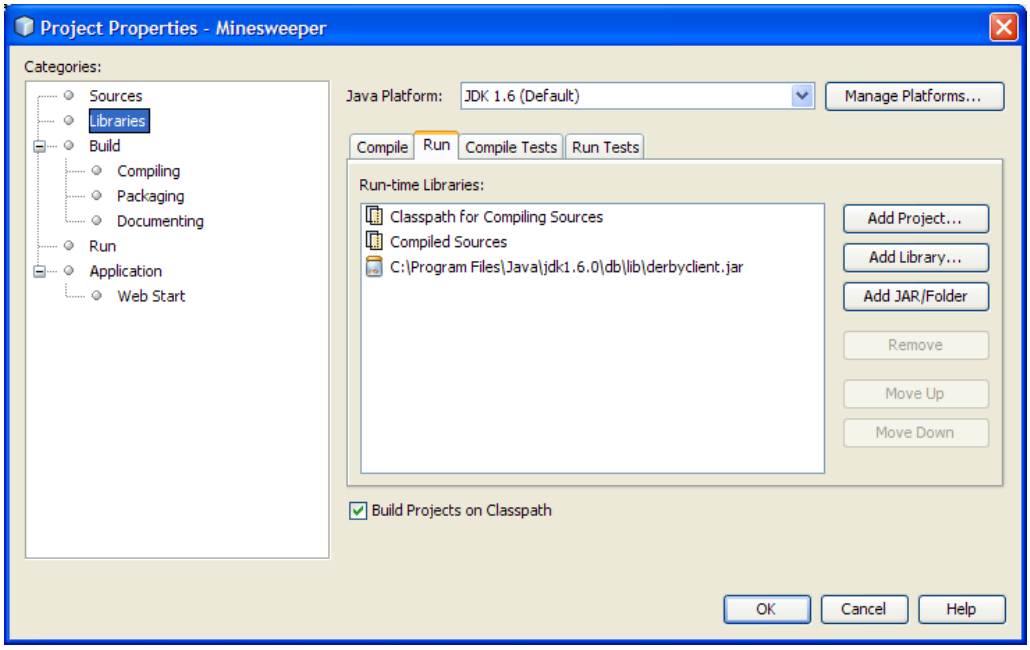
Poznámka: Názov databázy, meno a heslo používateľa musí zodpovedať hodnotám uvedeným v triede `DatabaseSetting`.

Úloha 0.94: Spustite databázový systém.

Zvoľte v kontextovom menu „Start Server” ak sa tak už nestalo.

Úloha 0.95: Pridajte do classpath ovládač pre prístup k databáze.

Zvoľte v kontextovom menu nad projektom v záložke „Projects” položku „Properties” a v dialógu „Libraries > Run” pridajte jar súbor s ovládačom `derbyclient.jar`, ktorý sa nachádza v inštalácii databázového servera Derby.



Úloha 0.96: Spustite hru **Minesweeper** a otestujte implementovanú funkčnosť.

Zdroje

[1] Zdrojové kódy:
- DatabaseSetting.java

Doplňujúce úlohy

Úloha 0.97: Rozšírte ukladanie časov hráčov o úroveň, v ktorej bol výsledok dosiahnutý (`BEGINNER`, `INTERMEDIATE`, `EXPERT`). Modifikujte triedu `BestTimes` ako aj databázovú tabuľku `player_time`.

Odporúčaná literatúra

[1] Cay S. Horstmann, Gary Cornell: Core Java, Volume 1: Fundamentals, 9th Edition. Prentice Hall PTR, December 2012, 1008 pp. ISBN 0137081898.

[2] Cay S. Horstmann, Gary Cornell: Core Java, Volume 2: Advanced Features, 9th Edition. Prentice Hall PTR, March 2013, 1152 pp. ISBN 013708160X.

[3] Pavel Herout: Učebnice jazyka Java, 5. vyd. Kopp, 386 pp. ISBN: 978-80-7232-398-2.

[4] The Java Tutorial http://docs.oracle.com/javase/tutorial

[5] J2SE reference documentation http://docs.oracle.com/javase/8/docs/api

[6] Hlavná stránka NetBeans IDE https://netbeans.org

[7] Java Code Conventions: http://www.oracle.com/technetwork/java/codeconvtoc-136057.html