Knihovna pro vizualizaci grafů v prostředí Java

Roman Vyčánek

Bakalářská práce
2018

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2017/2018

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Roman Vyčánek**
Osobní číslo:	**A15079**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Forma studia:	**prezenční**
Téma práce:	**Knihovna pro vizualizaci grafů v prostředí Java**
Téma anglicky:	**A Library for the Visualisation of Graphs Using Java Technology**

Zásady pro vypracování:

1. **Zpracujte stručný přehled řešené problematiky, včetně v současnosti používaných řešení.**
2. **Naprogramujte knihovnu pro vizualizaci grafů v prostředí Java. Knihovna bude schopna dynamického vykreslování ze zadaného XML formátu.**
3. **Sestavte přehledný manuál pro práci s výslednou knihovnou.**
4. **Vypracujte sadu ukázek praktického použití knihovny.**
5. **Výsledky práce vhodně prezentujte v prostředí Internetu.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **MATOUŠEK, Jiří a Jaroslav NEŠETŘIL. Kapitoly z diskrétní matematiky. 2. upr. vyd. Praha: Karolinum, 2000, 377 s. ISBN 80-246-0084-6.**
2. **Gephi: Features. [online]. [cit. 2013-03-21]. Dostupné z: https://gephi.org/features/**
3. **Itnetwork.cz [online]. [cit. 2017-11-28]. Dostupné z: https://www.itnetwork.cz/java**
4. **Tutorialspoint.com [online]. [cit. 2017-11-28]. Dostupné z: https://www.tutorialspoint.com/java_essential_training**
5. **BARTUŠKA, Karel a Emanuel SVOBODA. Fyzika pro gymnázia: molekulová fyzika a termika. 4. přeprac. vyd. Praha: Prometheus, 2000, 244 s. ISBN 80-7196200-7.**
6. **ČADA, Roman, Tomáš KAISER a Zdeněk RYJÁČEK. Diskrétní matematika. 1. vyd. Plzeň: Západočeská univerzita v Plzni, 2004, 170 s. ISBN 80-708-2939-7.**

Vedoucí bakalářské práce: **Ing. Bc. Pavel Vařacha, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce: **15. prosince 2017**

Termín odevzdání bakalářské práce: **25. května 2018**

Ve Zlíně dne 15. prosince 2017

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 22. 5. 2018

.....
podpis diplomanta

ABSTRAKT
Bakalářská práce se věnuje vizualizaci grafů v prostředí Java. Teoretická část představuje rešerši podobných řešení na Internetu. Dále jsou zde popsány použité technologie, které jsou dále využívány v praktické části. Praktická část popisuje vytvořenou knihovnu, a seznamuje uživatele s jejím dalším využitím pomocí manuálu a ukázek.

Klíčová slova: vizualizace grafů, Java, dynamická knihovna

ABSTRACT

Bachelor thesis deals with visualization of graphs in Java environment. The theoretical part presents similar programs on the Internet. Further, there are described the used technologies, which are further utilized in the practical part. The practical part describes the created library, and introduces users to its further use with manual and sample.

Keywords: graph visualization, Java, dynamic library

Poděkování

Tímto bych rád poděkoval svému vedoucímu panu Ing. Bc. Pavlu Vařachovi, Ph.D. za vedení diplomové práce a čas strávený při konzultacích.

OBSAH

ÚVOD

Cílem této bakalářské práce je návrh a následné vytvoření knihovny pro vizualizaci grafů v prostředí Java. Jedná se o grafy znázorňující vazby mezi různými entitami v podobě vrcholů a hran.

Motivací pro řešení této práce je možnost efektivního zapojení jejích výsledků do projektu „Inovace systémů řízení subjektů cestovního ruchu pomocí nástrojů procesního řízení“, jehož řešitelem je kromě dalších soukromých i akademických subjektů, také Univerzita Tomáše Bati ve Zlíně. Tento projekt je spolufinancován Technologickou agenturou České Republiky (TAČR) a jeho důležitou součástí bude i samostatná knihovna pro vykreslování grafů, která bude nahrazovat současné metody vykreslování pomocí finančně nákladného software ARIS [22].

Teoretická část obsahuje rešerši podobných programů na internetu a jejich srovnání s grafickou knihovnou. Dále je v teoretické části popsána historie programovacího jazyka JAVA, cíle, s kterými byl tento jazyk vyvíjen a nejpoužívanější IDE v současné době, Eclipse.

Následují definice hran a vrcholů, s různými tipy popisů grafu.

Praktická část rozebírá návrh grafické knihovny, popisuje jednotlivé třídy a rozebírá možnosti stylování grafu.

Součástí této knihovny jsou i algoritmy schopné rozložit vrcholy na ploše s ohledem na jejich vzájemné vazby tak, aby čitelnost grafu byla pro uživatele co možná nejnázornější. Jedná se o kruhové rozložení a organizační algoritmus využívající Hookův zákon popisující pružné deformace.

Praktická část taktéž obsahuje manuál vysvětlující implementaci knihovny do nového projektu Java. Jsou zde také uvedeny ukázky různého využití grafické knihovny.

Závěrečná část se věnuje prezentaci grafické knihovny v prostředí internetu prostřednictvím webu GitHub.

I. TEORETICKÁ ČÁST

1 PODOBNÉ PROGRAMY NA INTERNETU

Tato kapitola se věnuje již existujícím řešením na internetu, jejich vlastnostem a porovnání s grafickou knihovnou.

1.1 Gephi

Gephi [1] je multiplatformní open-source software distribuovaný pod licencemi CDDL 1.0 a GNU General Public License v3. Gephi má verze pro Mac OS X, Windows i Linux. Díky české lokalizaci, je práce pro uživatele jednoduchá.

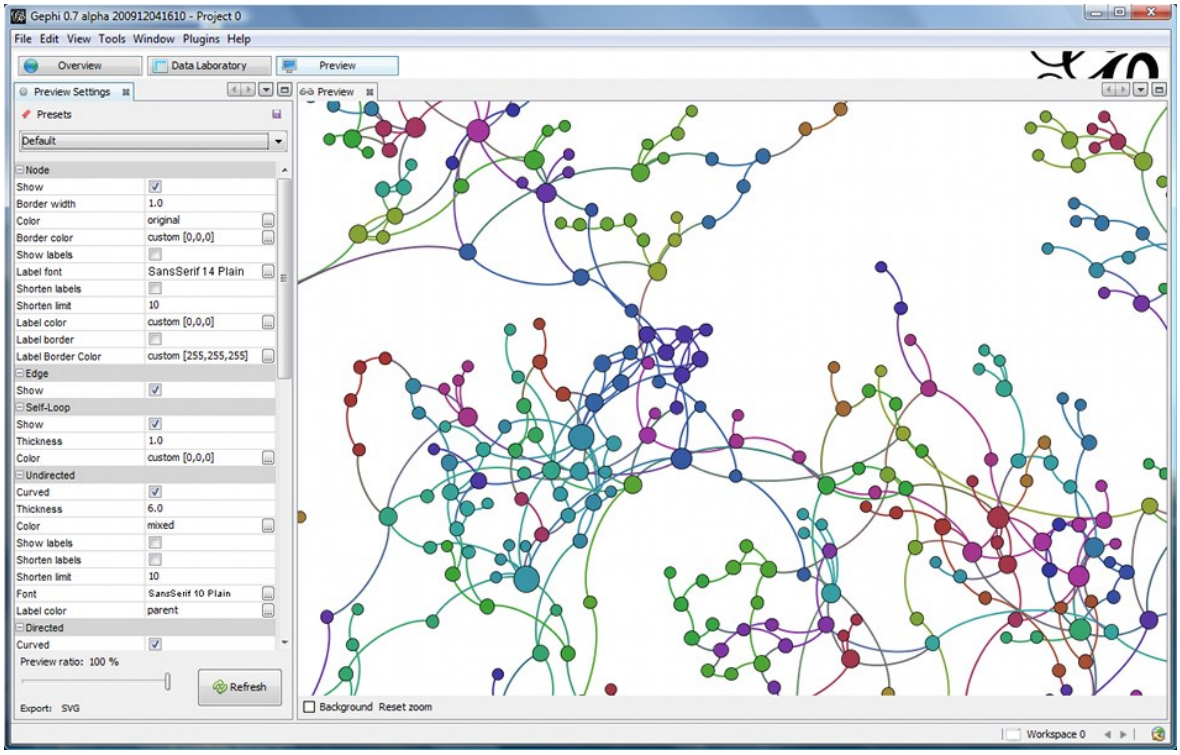
Hlavní vlastnosti

Gephi umožňuje víceúrovňové rozložení uzlů, lze také slučovat více uzlů do jednoho nebo třeba filtrovat zobrazení podle délky vazby. Z Matematických funkcí má například náhodné generátory, nebo funkci pro nalezení nejkratší možné cesty.

Možná je i dynamická analýza grafu, kdy uživatel má možnost sledovat, jak se síť vyvíjí v průběhu času. Nechybí ani možnost grafického přizpůsobení. Uživatel může měnit barvy velikosti, nebo přidávat popisky k jednotlivým uzlům. Formáty pro export jsou PDF, SVG nebo PNG. Data mohou být exportována ve formátu CSV kdy každý řádek je uzel nebo hrana.

Další výhodou je možnost přidávání pluginů, které si následně program sám udržuje aktuální.

Celý program je psán v programovacím jazyku Java a pro grafiku využívá OpenGL



Obrázek 1: Gephi aplikace [1]

1.2 yFiles for Java

YFiles [2] je rodina produktů pro široké použití na různých platformách, pracující s různými programovacími jazyky od Java(FX) po Microsoft .NET. Všechny varianty vypadají velmi podobně na různých operačních systémech díky automatickému algoritmu pro layout.

Hlavní vlastnosti

Zde jsou popsány hlavní výhody yFiles v uvedených oblastech [3]

- **Vykreslování dat**

  Po vypočtení rozložení grafů yFiles vytvoří animaci, která plynule přesune jednotlivé uzly i hrany na novou pozici. yFiles podporuje vykreslování detailů v závislosti na zoomu, díky čemuž jsou skryty prvky, které by byly nečitelné a při větším přiblížení je teprve začne zobrazovat. Křížení hran je vyřešeno mosty, díky kterým je jednodušší vidět kam každá hrana vede. Další možnost zvýšení čitelnosti je pomocí filtrování.

- **Vstupy a výstupy**

  yFiles podporuje ukládání a načítání grafů pomocí standartního formátu GraphML. Mimo jiné je možnost vytvářet diagramy pomocí strukturovaných dat jako například CSV, JSON, XML, databází a jiných. yFiles zvládá exportovat grafy jako rastrovou grafiku, nebo s pluginy je možný export jako PDF.

- **Automatické rozložení grafu**

  Poskytuje vysoce efektivní a uživatelsky modifikovatelné algoritmy pro automatické rozložení. Rozložení se propočítává podle různých kritérií. Možná rozložení jsou Hierarchické, stromové, kruhové a další. Přechody mezi různými rozloženími mohou být animovány.

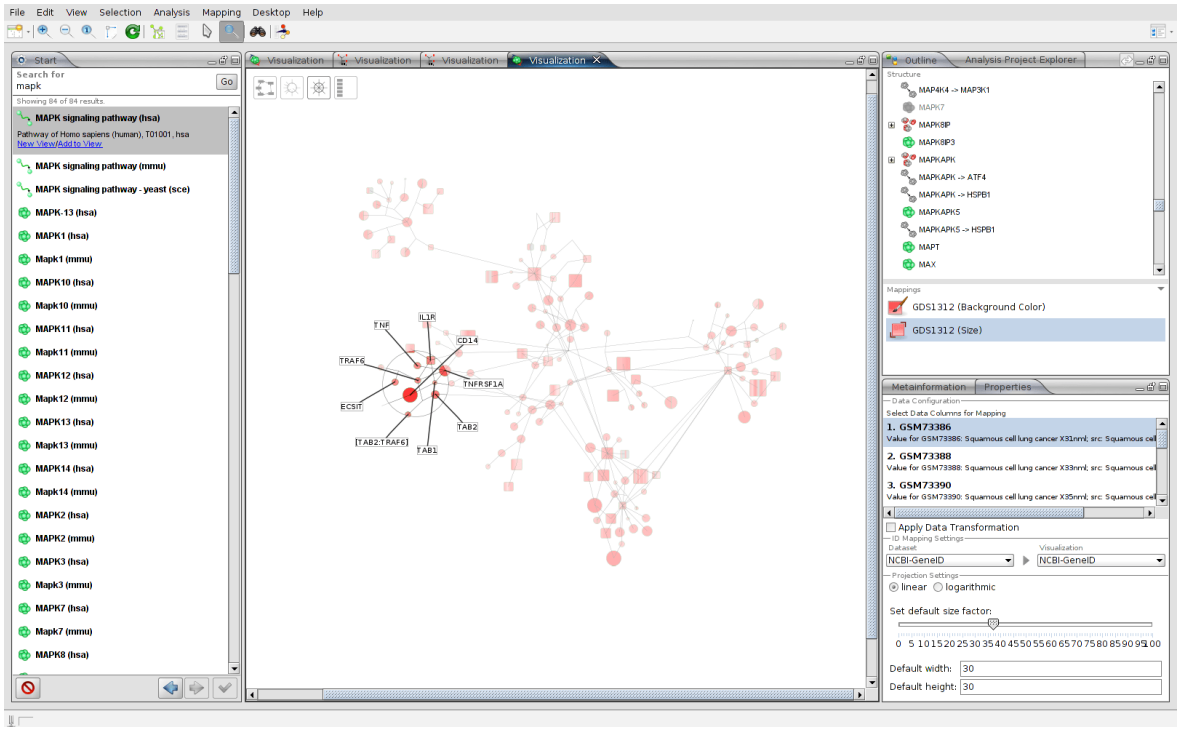
- **Algoritmus pro trasování hran**

  Je to algoritmus výpočtu dráhy hran. Cílem tohoto algoritmu je nalezení řešení, kdy dráhy neprocházejí přes vrcholy.

V následující tabulce jsou uvedeny ceny licencí [4]:

Tabulka 1: Ceny licencí

Název	Single Developer Licence	Project licence	Site Licence
Počet vývojářů	1 na jméno	3 floating	neomezeno
Počet aplikací	neomezeno	1	neomezeno
Počet webových stránek	neomezeno	neomezeno	1
Počet domén	1	2	4
Cena	12,870E	25,740E	59,400E



Obrázek 2: yFiles aplikace [2]

1.3 draw.io

Draw.io [5] je opensource projekt, který se zabývá vykreslováním vývojových diagramů, myšlenkových map, elektrických diagramů, nebo třeba i nákresů pokojů. Desktopové verze existují na Windows, macOS, Linux, i Crhome OS. Webová verze mají podporu pro všechny internetové prohlížeče.

Hlavní vlastnosti

Zde jsou uvedeny hlavní výhody využívání draw.io v uvedených oblastech.

Jednoduchý pro práci

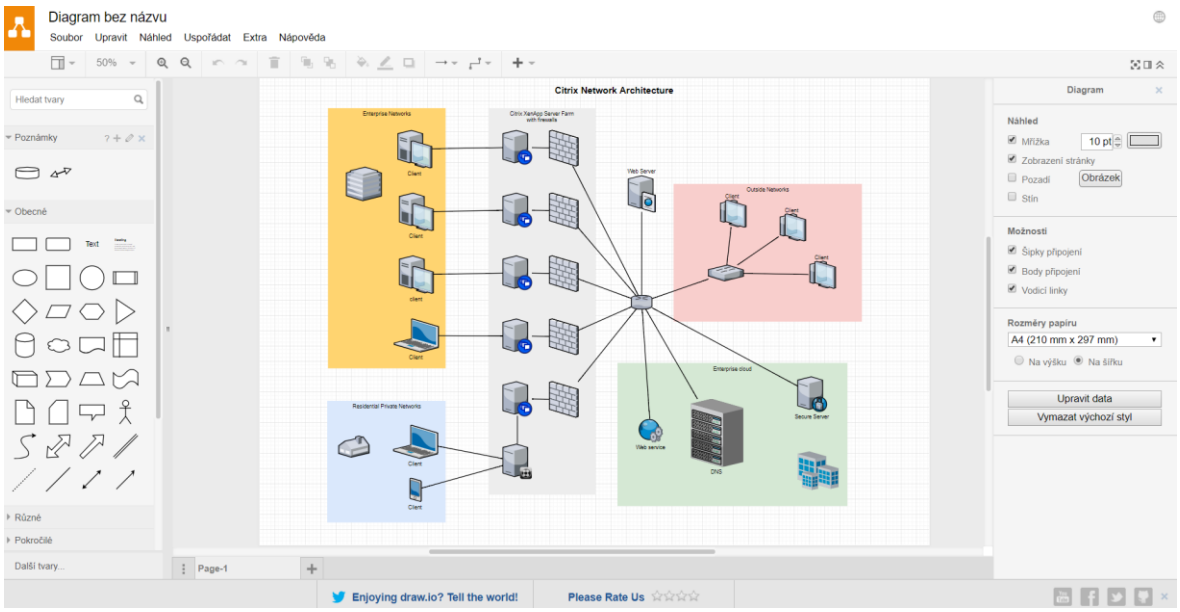
Draw.io má intuitivní interface, s podporou drag and drop. Výhodou je i webová aplikace, díky které není nutné nic instalovat. Navíc existuje s českou lokalizací.

Široká funkcionalita

Podporuje import z cloud úložišť, z CSV souborů, export je možný jako PNG nebo JPG obrázky, možný je i export do PDF nebo do XML. Uživatelé mohou pracovat jak na mobilních zařízení, tak na počítačích online i offline.

Bezpečnost a spolehlivost

Je navržen s možností škálování obsahu, s velmi optimalizovaným zdrojovým kódem. Aplikace neukládají data o uživatelích a uživatel vlastní obsah vytvořený touto aplikací.



Obrázek 3: drav.io aplikace [5]

1.4 Shrnutí

Všechny aplikace mají podobný základ, hlavní rozdíl je v tipu licencí. Knihovna si proto bere hlavní myšlenky a dále je rozvíjí.

2 JAVA

Java je univerzální multiplatformní programovací jazyk, který je objektově orientovaný a navržený tak, aby měl jednoduchou syntaxi.

Jeho cílem je umožnit vývojářům „napsat jednou, spustit kdekoli," což znamená, že kompilovaný kód Java může běžet na všech platformách, které podporují Javu bez nutnosti kompilace.

Java aplikace jsou většinou kompilovány do byte kódu, který může běžet na libovolném virtuálním stroji (Java Virtual Machine) bez ohledu na architekturu počítače. [6]

2.1 Historie Javy

1991 – James Gosling, Bill Joy, Mike Sheridan a Patrick Naughton započali Stealth Project (Tajný projekt), který měl za cíl vytvořit systém pro domácí spotřebiče. Projekt se později přejmenoval na Green Project (Zelený projekt), a tým vývojářů dostal název Green Team (Zelený tým). Vedoucím projektu se stal James Gosling a jazyk na kterém začali pracovat pojmenovali Oak (dub). [7; 8]

1995 – Přejmenování jazyka z Oak na Java. Java je americké slangové označení pro kávu.

1996 – Java Development Kit (JDK) 1.0

1998 – J2SE 1.2

Nejvýznamnější změny: knihovna Swing pro sestavení grafického rozhraní, Java 2D API, Java collections framework (JSR 166), Accessibility API, JDBC 2.0, Java Plug-in, Java IDL – pro podporu technologie CORBA, absolutní podpora Unicode včetně psaní v japonštině, čínštině a korejském jazyce, lepší výkon JIT-kompilátoru. [9]

16. 1. 2018 - Java SE 9.0.4

2.2 Hlavní cíle jazyka Java

Jazyk Java se řídí těmito cíli [10]:

Jednoduchý, objektově orientovaný

Díky jednoduchosti mohou být programátoři produktivní, od samého začátku, a objektové programování získává stále větší převahu mezi programátory.

Robustní a bezpečný

Poskytuje rozsáhlou kontrolu kompilace, po níž následuje další kontrola běhu.

Nezávislý na architektuře a přenositelný

Programy jsou spustitelné na většině zařízení díky tomu, že aplikace běží ve virtuálním stroji Java.

Vysoký výkon

Automatický garbage collector běží na pozadí a stará se o využití paměti, což vede k lepšímu výkonu.

interpretovaný, více vláknový a dynamický

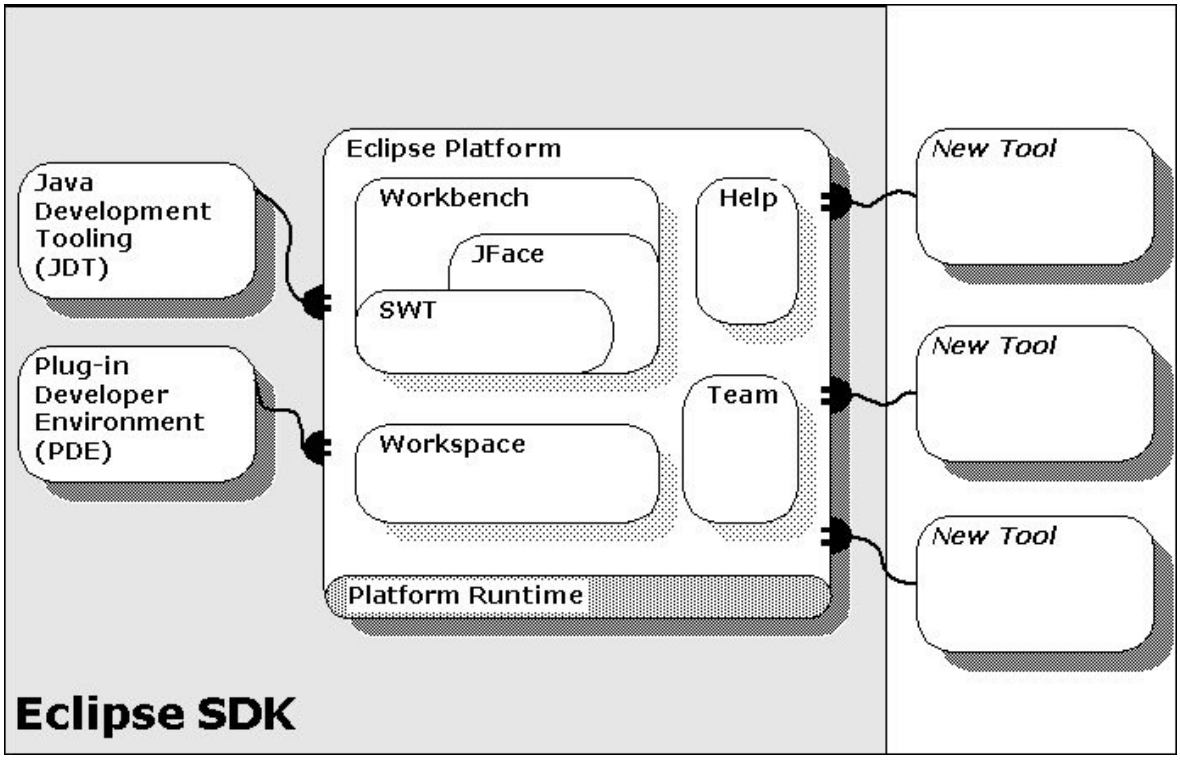
poskytuje třídu THREAD pro práci s více vláknovými aplikacemi

2.3 Vývojová prostředí pro Java IDE

„*IDE, Integrated Development Enviroment je programovým vybavením, které slouží vývojářům a programátorům, ve většině případů se zaměřením na určitý programovací jazyk“* [11]

2.3.1 Eclipse

Základní koncept, na kterém je postaven Eclipse je koncept plug-inů. Plug-iny jsou balíčky kódu které přidávají funkce do systému. Eclipse SDK obsahuje základní platformu a dva toolkity, Java development tools (JDT), které realizuje plnohodnotné vývojové prostředí Java a Plug-in Developer Environment (PDE), který přidává nástroje zjednodušující vývoj plug-inů a dalších rozšíření. [12]



Obrázek 4: Modularita Eclipse [12]

Eclipse je psaný v programovacím jazyce Java a je primárně používaný pro vývoj aplikací v Javě, ale díky plug-inům může být používán pro vývoj aplikací i v jiných programovacích jazycích, například Perl, PHP, Python, C, C++, C#, JavaScript a dalších.

Aktuální verze Eclipse je Oxygen 4.7 a v červnu má vyjít nová verze 4.8 s názvem Photon. [13]

Popis rozložení panelů

Rozložení panelů je volitelné, a každý uživatel si ho může přizpůsobit podle svých představ.

Package Explorer

Je to panel, ve kterém uživatel vidí všechny své projekty, každá třída musí být v balíku, který je ve složce src.

Editor window

Toto je okno, ve kterém se píší zdrojové kódy programu.

Outline

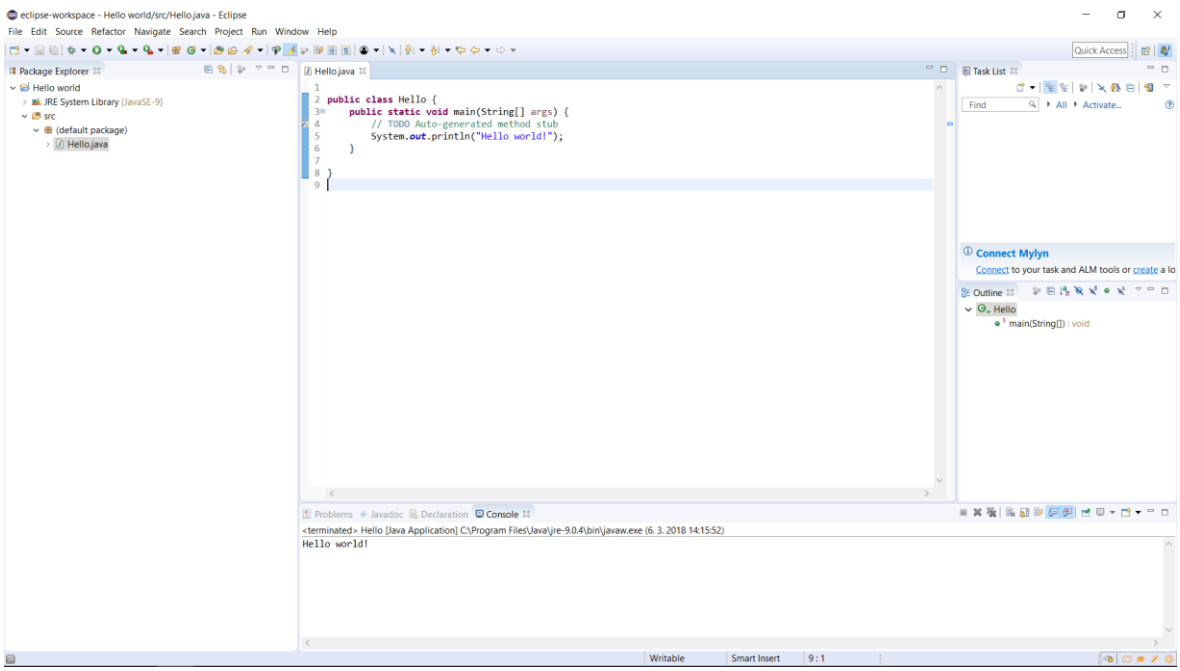
Je to panel, který slouží pro přehled proměnných, nebo metod pro danou třídu.

Tool bar

Je to panel s nejčastěji používanými funkcemi, jako jsou nový, uložit, spustit, testovat a další

Console

Eclipse má svou konzoli se kterou uživatel pracuje úplně standardně.



Obrázek 5: Vývojové prostředí Eclipse

3 ÚVOD DO GRAFŮ

Graf G je uspořádaná dvojice (V, H), kde V je neprázdná množina a H je množina dvoubodových podmnožin množiny V. Prvky množiny V se jmenují vrcholy grafu G a prvky množiny H hrany grafu G. [14]

$$G = (V, E) \tag{1}$$

3.1 Hrana

Jde obecně o dvojice vrcholů grafu. Nejčastěji je hrana znázorněna jako spojnice dvou vrcholů.

Dělení hran

Neorientovaná hrana: Jde o neuspořádanou dvojici vrcholů, hrana je průchozí v obou směrech.

Orientovaná hrana: Jde o uspořádanou dvojici vrcholů, hrana je průchozí pouze v jednom směru.

Smyčka: Hrana, jejíž počáteční i koncový vrchol je totožný.

Násobné hrany: Jsou hrany, které spojují tytéž vrcholy.

Typy grafů v závislosti na typech hran

Zde jsou uvedeny základní tipy grafů [15]:

Prostý graf je takový orientovaný nebo neorientovaný graf, v němž je násobnost každé hrany nejvýše rovna jedné.

Multigraf je graf který není prostý

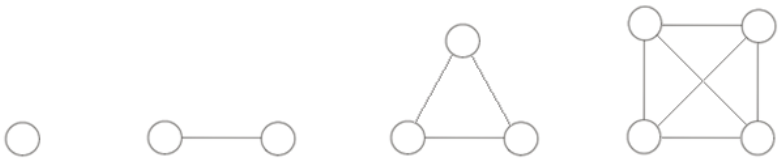
Jednoduchý graf je graf bez smyček

Diskrétní graf je graf, který neobsahuje hrany.

Orientovaný graf je graf obsahující orientované hrany

Úplný graf Je graf, v němž mezi každými dvěma vrcholy existuje právě jedna hrana.

Úplný graf, který má n vrcholů, má přesně $n(n-1)\ /2$ hran



Obrázek 6: Příklady úplných grafů

3.2 Vrchol

Vrchol, někdy též nazýván jako uzel je prvek z množiny vrcholů V. Nejčastěji se vrchol znázorňuje jako kruh.

4 REPREZENTACE GRAFŮ

„*Obecný popis grafu, bez ohledu na prostředek, který je použit, musí popisovat množinu vrcholů V, množinu hran E a incidenční zobrazení f. Jedině takovým popisem je graf zadán úplně.*“ [16]

4.1 Matice sousednosti

„*Nechť je dán obyčejný graf G = (V, H) s množinou hran H= {h_1, h_2, …, h_n} a množinou uzlů V = {v_1, v_2, …, v_m}. Matice sousednosti je čtvercová matice řádu m, kterou si označíme S.*“ [17] Prvky této matice s_{jk} jsou dány předpisem:

Pro neorientované grafy

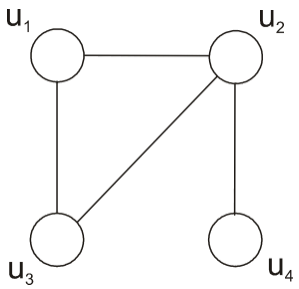
- s_{jk}=1, jestliže mezi uzly v_j a v_k existuje hrana
- s_{jk}=0, jestliže mezi uzly v_j a v_k hrana neexistuje

Pro orientované grafy

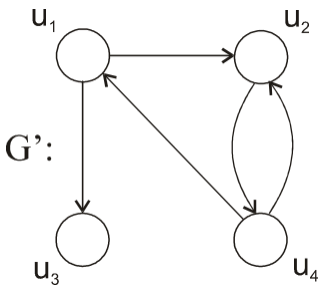
- s_{jk}=1, jestliže existuje orientovaná hrana z uzlu v_j do uzlu v_k
- s_{jk}=0, jestliže neexistuje orientovaná hrana z uzlu v_j do uzlu v_k

Vlastnosti matice sousednosti S:

- na hlavní diagonále jsou nuly (uvažujeme jen jednoduché grafy)
- je symetrická podle hlavní diagonály pro neorientované grafy
- u orientovaného grafu je počet jedniček dvojnásobkem počtu hran
- u neorientovaného grafu počet jedniček odpovídá počtu hran



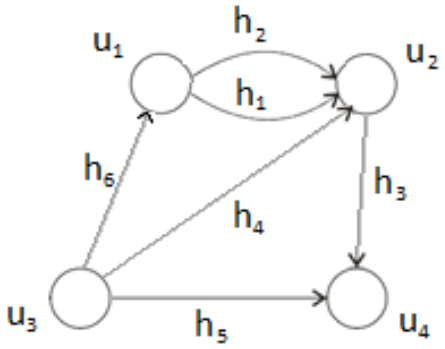
$$S = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

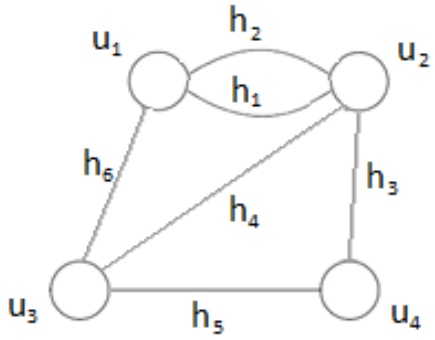


$$S' = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

4.2 Matice incidence

Má-li graf n vrcholů a m hran, matice incidende je obdélníková matice typu (m,n). Řádky odpovídají vrcholům, sloupce hranám grafu. U orientovaných grafů je v i-tém sloupci a j-tém řádku číslo 1, pokud je i-tý vrchol počátečním vrcholem j-té hrany, a číslo -1, pokud je jejím koncovým vrcholem. Ostatní prvky matice jsou nuly. Incidenční matice orientovaného grafu má v každém sloupci vždy jednu hodnotu 1 a jednu hodnotu -1. Ostatní jsou nuly. Neorientované grafy maní vždy v jednom sloupci právě dvě hodnoty 1 [18]





$$\begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccccc} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & -1 \\ -1 & -1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & -1 & 0 & -1 & 0 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccccc} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

4.3 Grafická reprezentace

Jde pro člověka o názornější reprezentaci grafu, kdy vrcholy a hrany jsou vykresleny do jednoduchého přehledného obrázku.

Touto reprezentací se budeme dále zabývat v praktické části, která bude o automatickém vykreslování grafů na základě různých typů vstupů. A jejich následném přehledném rozvržení na plátně.

II. PRAKTICKÁ ČÁST

5 STRUKTURA PROGRAMU

V této části je popsána struktura knihovny, její organizační algoritmy následně je zde i manuál pro práci s touto knihovnou. V poslední části, jsou uvedeny příklady využití knihovny.

5.1 Knihovna pro vizualizaci

Vrcholy

Pro vrcholy je vytvořena třída Vertex a tyto vrcholy jsou uloženy v privátním lineárním seznamu v třídě Graph.

Každý vrchol obsahuje základní informace o svém stavu:

- **Center** – proměnná typu Point a udává souřadnice vrcholu
- **backgroundColour** – proměnná typu Color a udává barvu vrcholu
- **radius** – proměnná typu int a udává poloměr vrcholu
- **name** – proměnná typu String a je popiskem u vrcholu
- **Velocity** – proměnná typu Point2D.Float, využívá se v organizačním algoritmu
- **netForce** – proměnná typu Point2D.Float, využívá se v organizačním algoritmu

Dále tato třída obsahuje getry a setry pro tyto proměnné, je implementována Override metoda equals(), která porovnává vrcholy podle jména a mnoho kombinací konstruktorů. Obsahuje také metodu drawVertex(Graphics g), která se volá při vykreslování grafu.

Hrany

Pro hrany je vytvořena třída Edge a tyto hrany jsou uloženy v privátním lineárním seznamu v třídě Graph.

Každá hrana obsahuje základní informace o svém stavu:

- **begin** – proměnná typu Vertex, odkazuje na počáteční vrchol
- **end** – proměnná typu Vertex, odkazuje na koncový vrchol
- **direction** – proměnná typu char, udává orientaci hrany
- **color** – proměnná typu Color, udává barvu hrany
- **arrowScale** – proměnná typu int, udává velikost šipek

Dále tato třída obsahuje getry a setry pro tyto proměnné, je implementována Override metoda equals(), která porovnává hrany podle počátečního o koncového vrcholu a mnoho kombinací konstruktorů. Obsahuje metodu drawEdge(Graphics g), která se volá při vykreslování grafu.

Typy orientací hran

Hrany mohou mít různou orientaci.

- „=“ – hrana není orientovaná
- „→“ a „←“ – hrana je orientovaná ve směru šipky
- „↔“ grafická šipka bude uvedena u obou konců hrany

Graf

Pro celý graf je vytvořena třída Graph.

Graf obsahuje základní informace o svém stavu:

- edges – proměnná typu LinkedList<Edge>, obsahuje všechny hrany grafu
- vertexes – proměnná typu LinkedList<Vertex>, obsahuje všechny vrcholy grafu
- parzer – proměnná typu Parzer, obsahuje metody usnadňující práci s textovými řetězci, také obsahuje metody pro import a export grafu do XML souboru
- distance – proměnná typu int, udává délku hran, využívá se při organizačním algoritmu
- windowSize – proměnná typu Point, využívá se při centrování grafů
- smoohting – proměnná typu boolean, ovládá antialiasing při vykreslování grafu

Dále tato třída obsahuje getry a setry pro proměnné. Metody pro přidávání a mazání vrcholů a hran, metodu graphToCenter() pro vycentrování grafu. Metodu organize(), která využívá pružných deformací k organizaci grafu. A metodu circleNodes(), přepočítávající polohy vrcholů do kružnice.

Metoda paintGraph(Graphics g), slouží pro vykreslení celého grafu. A metody toXML(String path) a metoda importXML(String path) využívající parzer pro ukládání grafu a jeho opětovnému načtení.

5.2 Algoritmy

Třída Graph obsahuje dva třídící algoritmy pro zpřehlednění grafického výstupu.

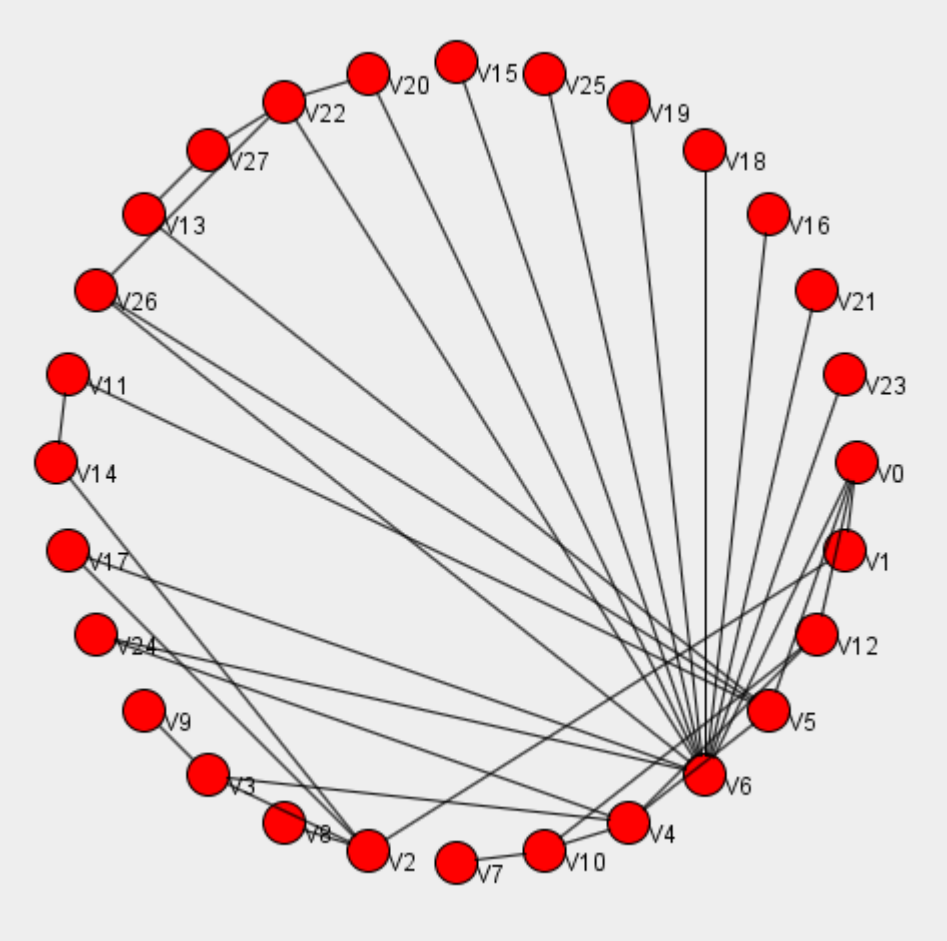
Kruhové zobrazení

Seřazení probíhá v jednom cyklu, který přiřazuje každému vrcholu souřadnice X a Y

$$X = radius * Sin(step * a * Pi / 180) \quad (2)$$

$$Y = radius * Cos(step * a * Pi / 180) \quad (3)$$

Kdy rádius je roven poloměru kružnice step udává velikost úhlu, o který jsou vždy dva vrcholy na kružnici posunuté, a udává o kolikátý vrchol jde.



Obrázek 7: Ukáza kruhového uspořádání

Organizační algoritmus

Pro tento algoritmus využíváme Hookův zákon [19], který popisuje pružné deformace. Organizační algoritmus probíhá ve dvou cyklech.

První cyklus postupně prochází jednotlivé vrcholy. Na začátku cyklu nastaví proměnou netForce pro daný vrchol na hodnoty [0,0]. Následují dva vnořené cykly. První cyklus prochází

všechny vrcholy, mimo vrcholu vybraného hlavním cyklem a vypočítává sílu těchto vrcholů, která působí na vrchol vybraný hlavním cyklem.

Síla se vypočítává pro každou souřadnici následovně:

$$newNetForce.x = oldNetForce.x + \frac{distance*(vertex.x - tempVertex.x)}{(vertex.x - tempVertex.x)^2 + (vertex.y - tempVertex.y)^2} \quad (4)$$

$$newNetForce.y = oldNetForce.y + \frac{distance*(vertex.y - tempVertex.y)}{(vertex.x - tempVertex.x)^2 + (vertex.y - tempVertex.y)^2} \quad (5)$$

Jak je vidět ze vzorců, důvodem vyloučení sebe sama je, že by tento případ vedl k pokusu dělení nulou.

Po tomto cyklu následuje druhý cyklus, který projde všechny vrcholy, které jsou spojeny hranou s vrcholem z hlavního cyklu. Tímto cyklem nastavuju sílu přímých vazeb.

$$newNetForce.x = oldNetForce.x + constant(tempVertex.x - vertex.x) \quad (6)$$

$$newNetForce.y = oldNetForce.y + constant(tempVertex.y - vertex.y) \quad (7)$$

Tímto byly vypočítaná síly působící na vrchol a v dalším kroku hlavního cyklu je vypočítána rychlost daného vrcholu.

$$newVelocity.x = damping(oldVelocity.x + NetForce.x) \quad (8)$$

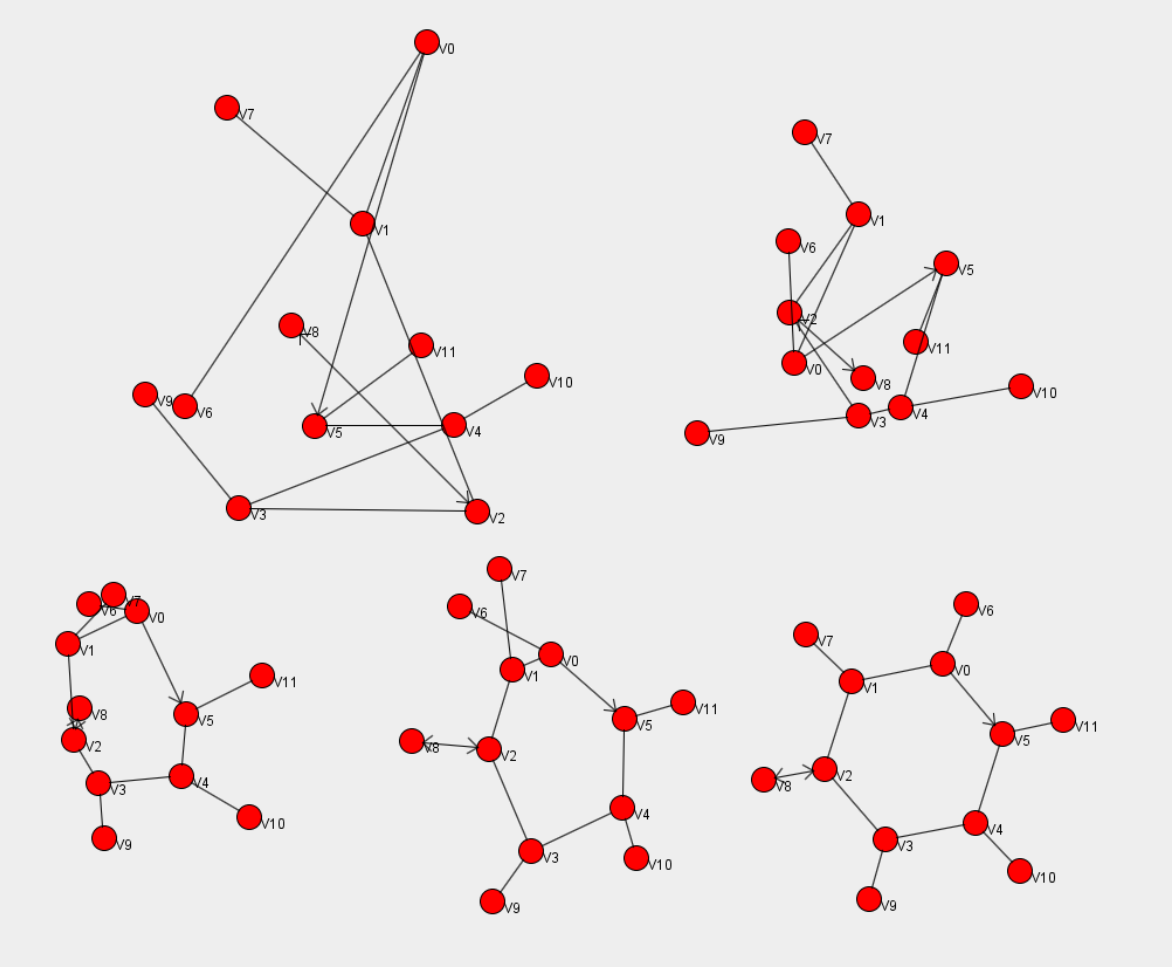
$$newVelocity.y = damping(oldVelocity.y + NetForce.y) \quad (9)$$

Na konci hlavního cyklu byla vypočítána rychlost pro všechny vrcholy grafu následuje druhý cyklus, který přičítá rychlost každého vrcholu k jeho pozici.

$$newPosition.x = oldPosition.x + velocity.x) \quad (10)$$

$$newPosition.y = oldPosition.y + velocity.y) \quad (11)$$

Tento algoritmus neuspořádá graf v jedné iteraci. Proto je tento algoritmus void metodou organize() volán dvěstěkrát. Kdy se vrcholy dvěstěkrát posunou do stabilnější pozice.



Obrázek 8: Ukázka postupné organizace vyrcholů

5.3 Stylování

Pro stylování se využívají metody z jednotlivých tříd, kterými lze upravovat velikost a barvu vrcholů, nebo zapínání antialiasingu. Pro složitější úpravy je vhodné využívat import grafu uloženého v XML.

Import grafu ze XML

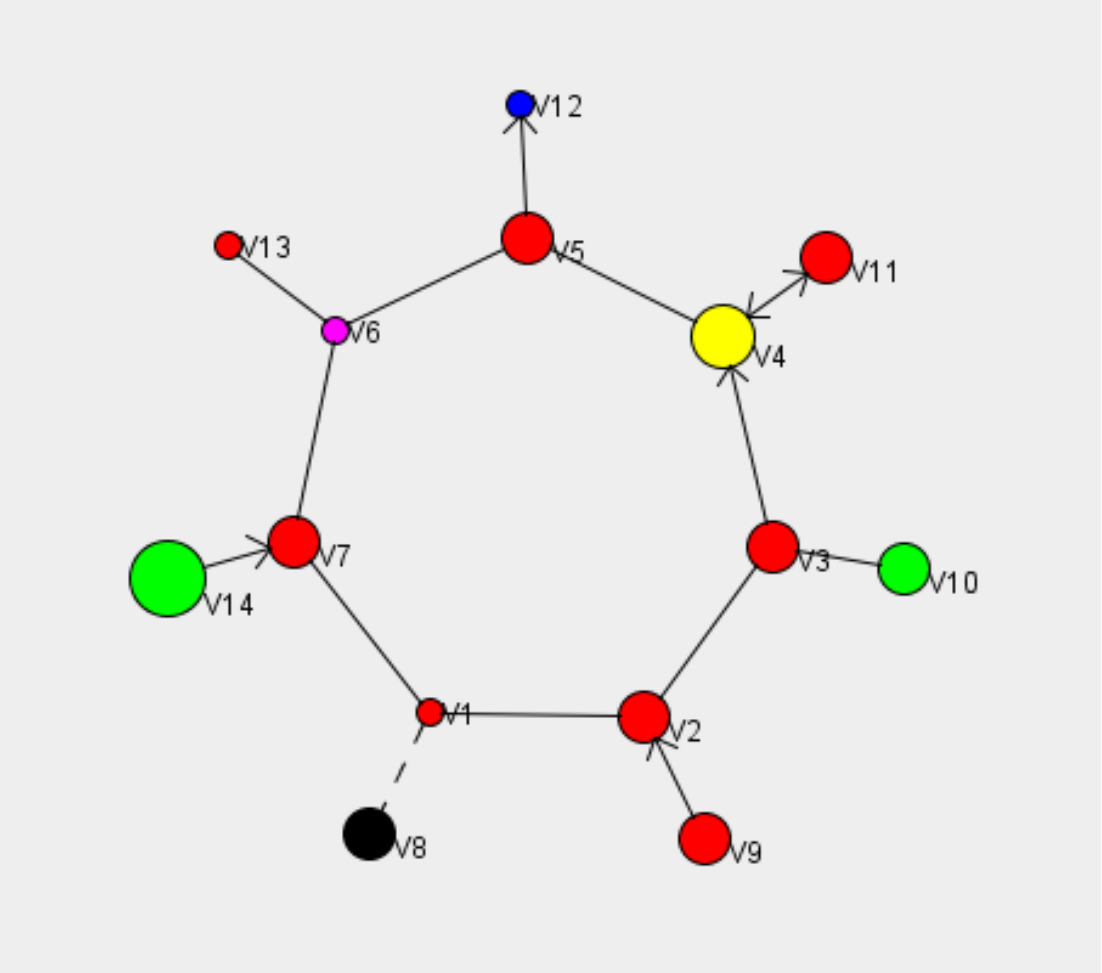
Import grafu ze XML je základní funkcí této knihovny. Tento soubor obsahuje informace o hranách a vrcholech grafu. Včetně jejich stylování.

Vstupní soubor odpovídající návrhu:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Graph>
    <Edges>
        <Edge ID="V2=V1" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V2=V3" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V4←V3" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V4=V5" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V6=V5" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V6=V7" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V1=V7" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V1=V8" RGB="0, 0, 0" dashed="true" scale="10"/>
        <Edge ID="V2←V9" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V3=V10" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V4↔V11" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V5→V12" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V6=V13" RGB="0, 0, 0" dashed="false" scale="10"/>
        <Edge ID="V7←V14" RGB="0, 0, 0" dashed="false" scale="10"/>
    </Edges>
    <Vertexes>
        <Vertex ID="V2" RGB="255, 0, 0" X="287" Y="339" radius="20"/>
        <Vertex ID="V1" RGB="255, 0, 0" X="204" Y="342" radius="10"/>
        <Vertex ID="V3" RGB="255, 0, 0" X="340" Y="269" radius="20"/>
        <Vertex ID="V4" RGB="255, 255, 0" X="317" Y="180" radius="25"/>
        <Vertex ID="V5" RGB="255, 0, 0" X="239" Y="142" radius="20"/>
        <Vertex ID="V6" RGB="255, 0, 255" X="165" Y="185" radius="10"/>
        <Vertex ID="V7" RGB="255, 0, 0" X="143" Y="267" radius="20"/>
        <Vertex ID="V8" RGB="0, 0, 0" X="174" Y="387" radius="20"/>
        <Vertex ID="V9" RGB="255, 0, 0" X="312" Y="389" radius="20"/>
        <Vertex ID="V10" RGB="0, 255, 0" X="394" Y="278" radius="20"/>
        <Vertex ID="V11" RGB="255, 0, 0" X="362" Y="150" radius="20"/>
        <Vertex ID="V12" RGB="0, 0, 255" X="241" Y="92" radius="10"/>
        <Vertex ID="V13" RGB="255, 0, 0" X="121" Y="150" radius="10"/>
        <Vertex ID="V14" RGB="0, 255, 0" X="86" Y="277" radius="30"/>
    </Vertexes>
</Graph>
```

Jak můžeme vyčíst soubor XML obsahuje čtrnáct vrcholů a mezi nimi hrany. Každý vrchol má v parametrech své jednoznačné jméno. Další parametry jako je pozice velikost a barva jsou volitelné a parzer je nevyžaduje.

Vrcholy jsou propojeny hranami, definovanými v sekci Edges a zde je výsledný graf vykreslený pomocí grafické knihovny



Obrázek 9: Grafický výstup z XML vstupu

6 MANUÁL PRO PRÁCI S KNIHOVNOU

Tato část popisuje, jak správně využívat grafickou knihovnu. Budou zde uváděny i ukázky v jazyce Java. V tomto manuálu budeme tvořit třídu Example pracující s grafickou knihovnou. Pro naučení základů programovacího jazyka JAVA využijte internetové učebnice. [20; 21]

6.1 Přidání knihovny do projektu

Zde je popsáno přidání knihovny graph do projektu ve vývojovém prostředí Eclipse. V nastavení projektu properties (Alt + Enter) → Java Build Path → Libraries → Modulepath → Add External JARs…, po nalezení příslušné knihovny se import potvrdí tlačítkem Apply and Close.

Obsah knihovny se dále importuje následovně:

```
import graph.Graph;
```

6.2 Otevření okna pomocí swing knihoven

Třída Example, rozšiřuje třídu JComponents. Třída Example obsahuje proměnou JFrame. Tato proměnná je grafickým oknem se kterým se bude dále pracovat. V konstruktoru se nastavuje velikost komponenty, ve které bude grafický výstup, tuto komponentu budeme v main funkci přidávat do proměnné frame.

V main funkci, je vytvořena komponenta example, která je následně přidána do okna. Poté je přizpůsobena velikost okna podle velikosti komponent a okno je nastaveno na viditelné.

```
package examples;

import java.awt.BorderLayout;
import java.awt.Dimension;
import javax.swing.JComponent;
import javax.swing.JFrame;


public class example01 extends JComponent {
    static int scalex = 500;
    static int scaley = 500;
    /**
         *
         */
    private static final long serialVersionUID = 4961943303303534846L;

    private static JFrame frame;

    public static void main(String[] args) {

        final example01 comp = new example01();
        comp.setPreferredSize(new Dimension(scalex, scaley));
        frame = new JFrame();
        frame.setBounds(100, 100, scalex, scaley);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(new BorderLayout(0, 0));
        frame.getContentPane().add(comp, BorderLayout.CENTER);

        frame.pack();
        frame.setVisible(true);

    }

}
```

6.3 Deklarace a inicializace proměnné

Proměnnou typu Graph, přidáme do třídy Example, tato proměnná bude dostupná v celé třídě Example.

```
static int scalex = 500;
static int scaley = 500;
/**
     *
     */
private static final long serialVersionUID = 4961943303303534846L;

// vytvoření a inicializace objektu
private final static Graph graph = new Graph();

private static JFrame frame;
```

6.4 Nastavení grafu

Graf po inicializaci má vypnuté vyhlazování hran, také je nutné nastavit délku hrany pro organizační algoritmus.

```
// povolení vyhlazování
graph.setAntialiasing(true);
// nastavení optimální délky hrany
graph.setDistance(250);
frame.pack();
frame.setVisible(true);
```

6.5 Vykreslení grafu

Pro vykreslení grafu je potřeba přidat metodu graph.paintGraph(Graphics g) do funkce která vykresluje komponentu example. To zajistíme voláním této metody v metodě komponenty paintComponent(Graphics G)

```
@Override
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    // vykreslení grafu
    graph.paintGraph(g);
}
```

6.6 Přidávání vrcholů a hran do grafu

Pro vytváření grafu existují dvě metody, první je pomocí textových definic hran a vrcholů. Nebo druhá metoda, která je složitější na zápis, ale zajišťuje široké možnosti individualizace grafu. Stylování je věnována část v kapitole 5.3. Následně se vrcholy seřadí organizačním algoritmem.

```
// vytvoření vrcholů a hran
graph.addEdges("V0=V1,V1=V2,V2=V3,V0=V3,V0←V4");

// organizace
graph.organize();
```

6.7 Individuální modifikace objektů

Pro nalezení jednotlivých objektů se používají metody getLineByID(String ID) a getVertexByID(String ID). Jednotlivé objekty se následovně upravují setry.

```
graph.getVertexByID("V4").setBackgroundColour(Color.GREEN);
// změna barvy hrany
graph.getEdgeByID("V4=V0").setColor(Color.red);
```

6.8 Zdrojový kód

```
package examples;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Graphics;

import javax.swing.JComponent;
import javax.swing.JFrame;

import graph.Graph;

public class example01 extends JComponent {
    static int scalex = 500;
    static int scaley = 500;
    /**
         *
         */
    private static final long serialVersionUID = 4961943303303534846L;

    // vytvoření a inicializace objektu
    private final static Graph graph = new Graph();

    private static JFrame frame;

    public static void main(String[] args) {

        final example01 comp = new example01();
        comp.setPreferredSize(new Dimension(scalex, scaley));
        frame = new JFrame();
        frame.setBounds(100, 100, scalex, scaley);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.getContentPane().setLayout(new BorderLayout(0, 0));
        frame.getContentPane().add(comp, BorderLayout.CENTER);

        // povolení vyhlazování
        graph.setAntialiasing(true);
        // vytvoření vrcholů a hran
        graph.addEdges("V0=V1,V1=V2,V2=V3,V0=V3,V0<V4");
        // nastavení optimální délky hrany
        graph.setDistance(250);
        // změna barvy Vertexu
        graph.getVertexByID("V4").setBackgroundColour(Color.GREEN);
        // změna barvy hrany
        graph.getEdgeByID("V4=V0").setColor(Color.red);
        // organizace
        graph.organize();

        frame.pack();
        frame.setVisible(true);

    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        // vykreslení grafu
        graph.paintGraph(g);
    }

}
```

6.9 Dostupné možnosti

WindowSize

Pomocí getrů a setrů nastavujeme velikost okna, pomocí tohoto parametru grafická knihovna zná střed plátna. Změnou tohoto parametru je tedy možné pohybovat plátnem.

setAntialiasing

díky této metodě je možné zapnout, popřípadě vypnout vyhlazování hran při vykreslování objektů na plátno.

addEdges

je metoda, která přijímá mnohonásobný textový vstup, který následně projde parzerem. Jednotlivé hrany se zadávají oddělené čárkou. Duplicitní hrany, jsou přeskočeny.

addVertexes

tato metoda je obdobou addEdges, slouží pro přidání vrcholů pomocí textového řetězce.

Reset

Tato metoda smaže všechny vrcholy a hrany.

getVertexByID

tato metoda přijímá jako vstupní parametr textový řetězec, a její návratová hodnota je null, pokud zadaný vrchol knihovna neobsahuje, nebo vrací vrchol se stejným ID.

getEdgeByID

tato metoda přijímá jako vstupní parametr textový řetězec, a její návratová hodnota je null, pokud zadanou hranu graf neobsahuje, jinak vrací hranu s hledanými vlastnostmi.

organize

metoda může být volána s parametrem, který udává počet iterací organizačního algoritmu. Nebo bez parametru, kdy se organizační algoritmus provede dvěstěkrát.

Distance

Pomocí getrů a setrů, se nastavuje délka hran, kterou využívá organizační algoritmus.

GraphToCenter

Tato metoda vycentruje celý graf na střed plátna, při kombinaci s nastavením WindowSize je možno těchto dvou metod využít pro posun grafu po plátně.

circleNodes

tato metoda slouží pro kruhovou organizaci grafů. Zavoláním se vrcholy rozmístí kolem středu plátna. Poloměr kružnice je udávám parametrem distance.

toXML

tato metoda vygeneruje XML soubor obsahující data z aktuálně vytvořeného grafu. Její parametr je cesta, kam se má soubor uložit

importXML

tato metoda načte graf ze souboru XML, nacházející se na adrese zadané v parametru.

isThereVertex

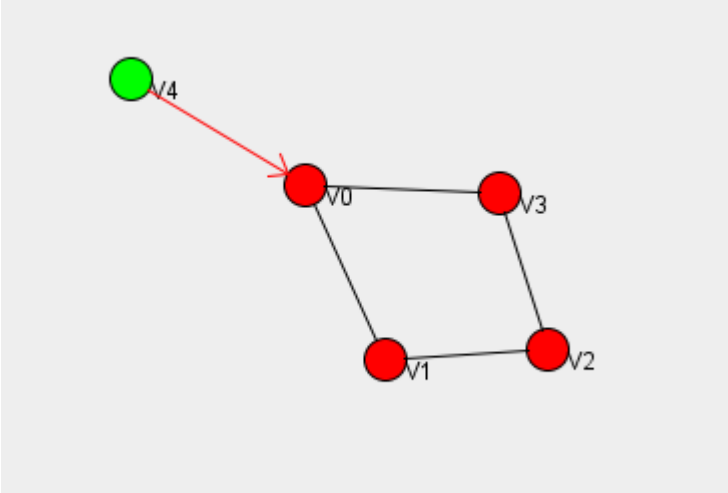
této metodě se v parametru předají souřadnice, a pokud tyto souřadnice leží v nějakém vrcholu vrací příslušný vrchol, jinak vrací hodnotu null.

7 PŘEHLED UKÁZEK

V této kapitole budou rozebrány ukázky vytvořené pomocí této knihovny

7.1 Manuál

Zdrojový kód k této ukázce je již rozebrán v kapitole 6.

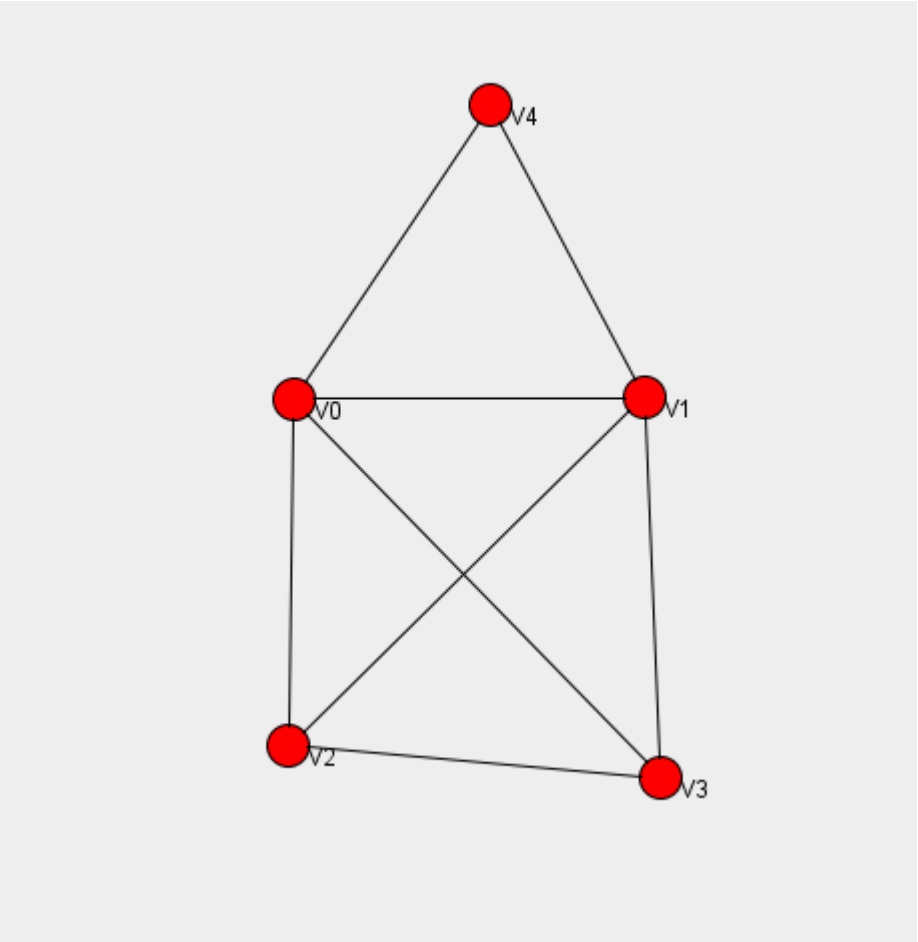


Obrázek 10: Grafický výstup manuálu

V grafickém výstupu lze vidět jak vrcholy s výchozí červenou barvou, tak i vrchol s modifikovanou barvou, taktéž hrana z vrcholu V4 do vrcholu V0 má změněnou barvu.

7.2 Ukázka 1

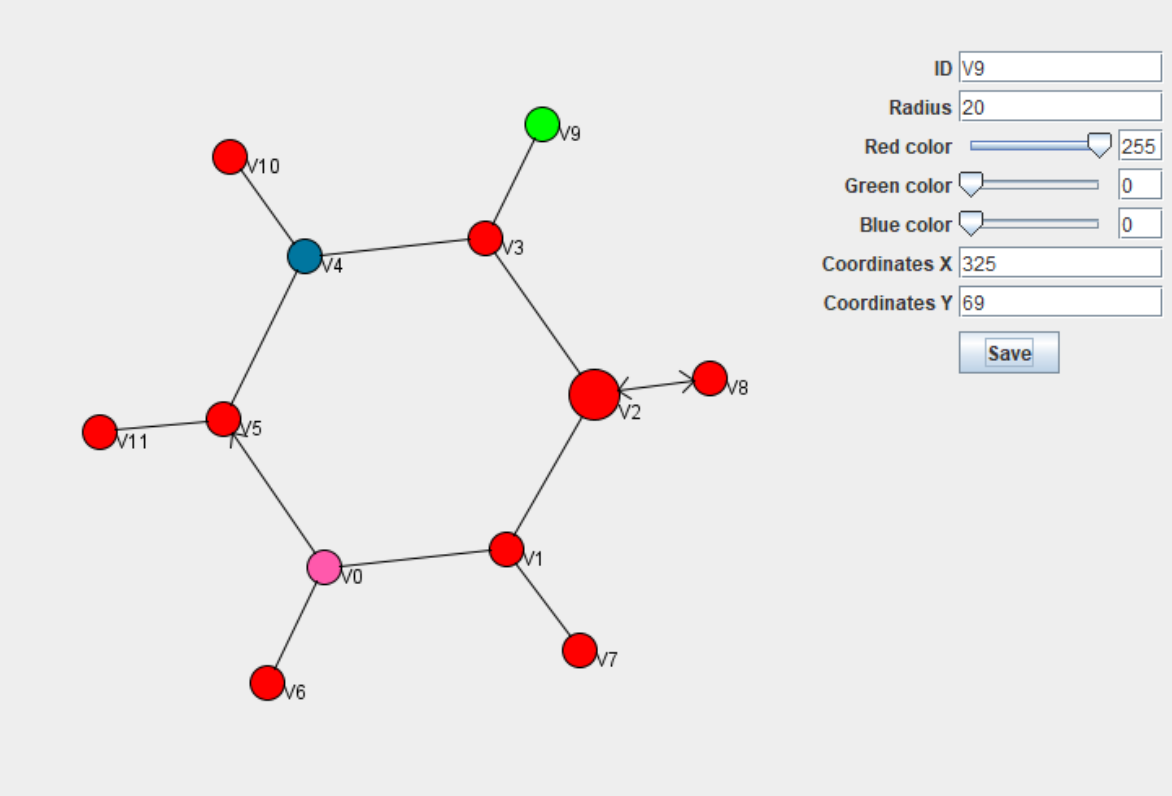
Tato ukázka představuje využití knihovny při obsluze událostí spojených s MouseListenerem, kdy pomocí modifikátorů knihovna přidává vrcholy na pozici kliku (klávesa Alt), vrchol z pozice kliku odebírá, nebo je možno po vybrání jednoho vrcholu bez modifikátoru, tento vrchol spojit s jiným vrcholem pomocí držení klávesy Shift a kliku na příslušný vrchol.



Obrázek 11: Grafický výstup ukázky 1

7.3 Ukázka 2

Tato ukázka využití knihovny představuje možný návrh modifikačního panelu, tento panel je obsluhován taky funkcemi z grafické knihovny. Vybrání specifického vrcholu se provádí podobně jako v první ukázce pomocí MouseListeneru.

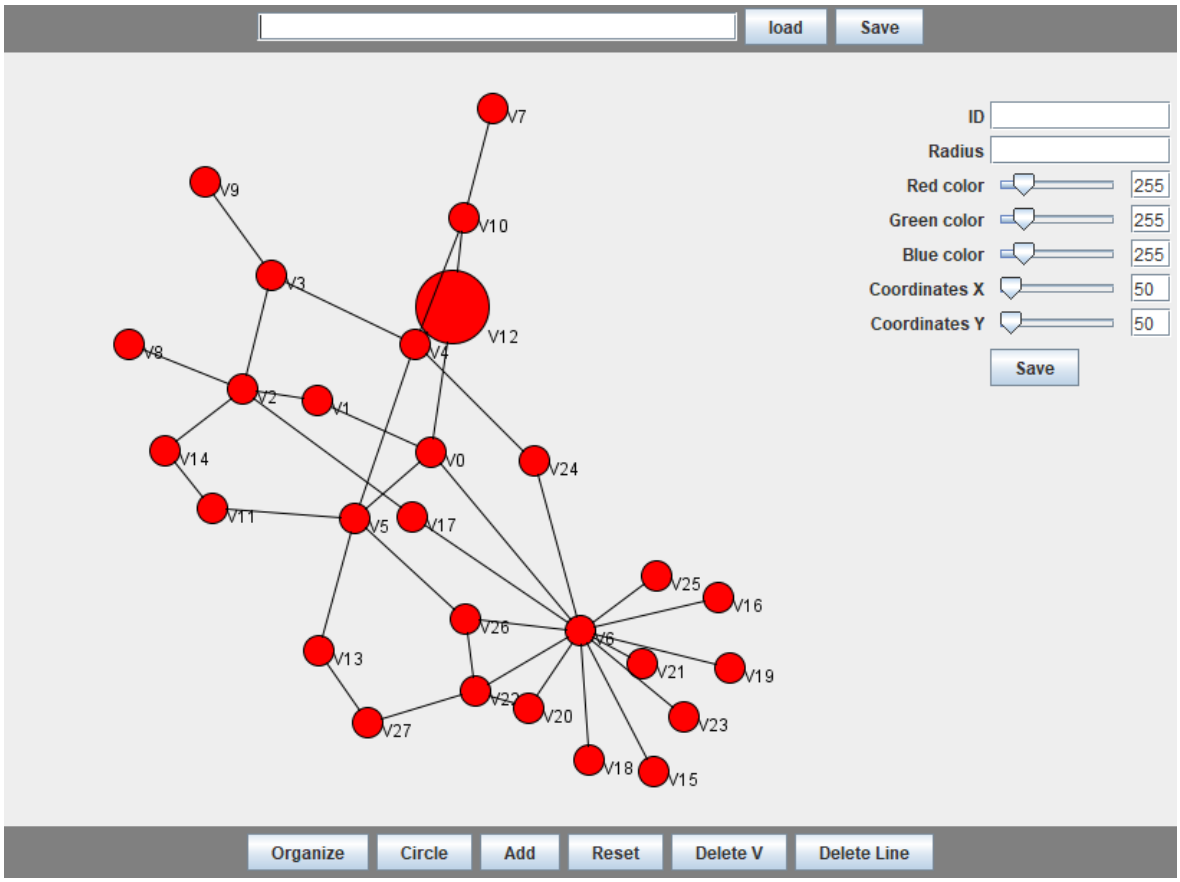


Obrázek 12: Grafický výstup ukázky 2

7.4 Ukázka 3

Tato ukázka kombinuje možnosti z první i druhé ukázky, takže je možné přidávat nebo odstraňovat vrcholy pomocí myši a modifikátorů. Tlačítka Organize a Circle uspořádávají graf do vizuálně přehledného rozložení. Tlačítka Load a Save slouží pro export a import grafu do XML souboru. tlačítko Add slouží pro přidávání nových vrcholů a orientovaných hran. Tlačítka Delete V a Delete Line, mažou zadané vrcholy, nebo hrany.

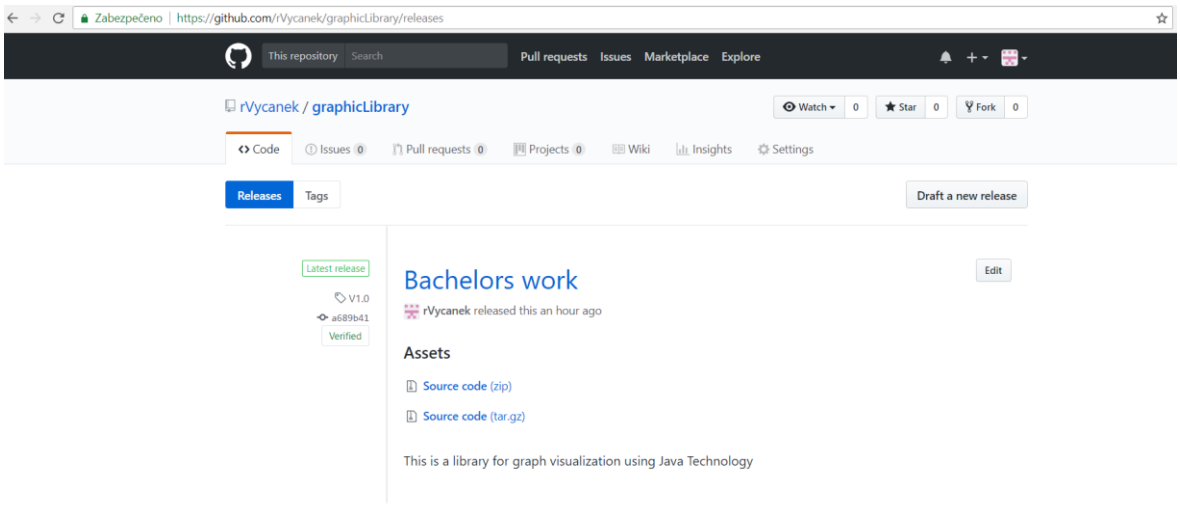
Tato ukázka by se dala považovat už za editační nástroj využívající tuto knihovnu k práci s grafy.



Obrázek 13: grafický výstup ukázky 3

8 PREZENTACE V PROSTŘEDÍ INTERNETU

Pro zpřístupnění knihovny široké veřejnosti byla knihovna vystavena na mém účtu GitHubu. Je veřejně přístupná a volně stažitelná.



Obrázek 14: prezentováno na webu Github.com

ZÁVĚR

Bakalářská práce se v úvodu věnuje současně existujícím řešením, uvádí příklady placeného programu i programů volně ke stažení. Dále následuje část, která se věnuje programovacímu jazyku JAVA, v této části je stručně popsána historie tohoto programovacího jazyku i popis nejpoužívanějšího vývojového prostředí Eclipse. Poté se čtenář seznámí s teorií grafů a možnostmi reprezentace grafů.

Praktická část popisuje strukturu grafické knihovny, hlavní část je věnována organizačnímu algoritmu, který využívá představu, že všechny vrcholy na sebe působí odpudivými silami, a všechny hrany se dají zjednodušit jako pružiny, které brání v pohybu vrcholů. Tyto pružiny působí opačnou silou a při vyrovnání působících sil se považuje graf za optimálně rozmístěný. Druhý způsob organizace je rozmístění vrcholů do kružnice, tento výpočet je matematicky méně náročný a je proveden po jednom cyklu.

Pro definici grafů jsou implementovány dvě základní možnosti vstupu. První možnost je pomocí textového řetězce, kdy se vstup zadává jako textový řetězec ve formátu ID prvního vrcholu + tip hrany + ID druhého vrcholu. Knihovna, pokud daný vrchol neexistuje, tak vrchol s požadovaným ID vygeneruje. V případě, že existují oba vrcholy, knihovna je spojí požadovanou hranou. Druhý způsob je pomocí XML souborů, které můžou obsahovat mnoho modifikátorů jak k hranám, tak i k vrcholům.

V závěru práce obsahuje podrobný manuál pro práci s knihovnou, v manuálu jsou popsány jednotlivé metody, které může uživatel využívat, tak i ukázky využití knihovny.

Případný budoucí rozvoj knihovny by se mohl věnovat přidání dalších organizačních algoritmů, optimalizaci trasování hran, přidání další třídy spravující grafický vzhled jednotlivých objektů a následné implementaci dědění grafického vzhledu z rodiče na potomky. Organizačnímu algoritmu by velmi pomohl přechod z dvojrozměrného prostoru do třírozměrného, který by umožnil další možný pohyb vrcholů. Stav, kdyby všechny vygenerované vrcholy ležely v jedné rovině, popřípadě na jedné přímce, je velmi nepravděpodobný.

Hlavní využití této knihovny bude její následná implementace do projektu „Inovace systémů řízení subjektů cestovního ruchu pomocí nástrojů procesního řízení“ řešeného s přispěním TAČR.

SEZNAM POUŽITÉ LITERATURY

[1] Gephi: Features. [Online] [cit. 2018-02-14] Dostupné z: https://gephi.org/features/.

[2] Yworks the diagram company. [Online] [cit. 2018-02-14] Dostupné z: https://www.yworks.com/products/yfiles-for-java-2.x.

[3] Gephi: Features. [Online] [cit. 2018-02-14] Dostupné z: https://www.yworks.com/products/yfiles/features.

[4] Yworks pricing. [Online] [cit. 2018-02-14] Dostupné z: https://www.yworks.com/products/pricing#info.

[5] draw.io. [Online] [cit. 14. 4 2018.] Dostupné z: https://about.draw.io/.

[6] IT slovník. [Online] [cit. 2018-03-27] Dostupné z: https://it-slovnik.cz/pojem/java/?utm_source=cp&utm_medium=link&utm_campaign=cp.

[7] Novotný, Luděk. Historie a vývoj jazyka Java. [Online] [cit. 2018-03-27] Dostupné z: https://www.fi.muni.cz/usr/jkucera/pv109/2003p/xnovotn8.htm.

[8] Faltýnek, Lukáš. linuxexpres.cz. [Online] [cit. 2018-03-27] Dostupné z: https://www.linuxexpres.cz/praxe/java-dnes-pri-salku-dobre-kavy.

[9] SUN DELIVERS NEXT VERSION OF THE JAVA PLATFORM. [Online] [cit. 2018-03-28] Dostupné z: http://web.archive.org/web/20070816170028/http:/www.sun.com/smi/Press/sunflash/1998-12/sunflash.981208.9.xml.

[10] oracle.com. Design Goals of the Java TM Programming Language. [Online] [cit. 2018-03-27] Dostupné z: http://www.oracle.com/technetwork/java/intro-141325.html.

[11] it-slovnik.cz. [Online] [cit. 2018-03-27] Dostupné z: https://it-slovnik.cz/pojem/ide.

[12] eclipse.org. Eclipse documentation. [Online] [cit. 2018-03-27] Dostupné z: https://help.eclipse.org/mars/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fguide%2Fua_help_setup_infocenter.htm.

[13] wiki.eclipse.org. Simultaneous Release. [Online] [cit. 2018-03-27] Dostupné z: https://wiki.eclipse.org/Simultaneous_Release.

[14] Jiří Matoušek, Jaroslav Nešetřil.: Kapitoly z diskrétní matematiky. Praha: Nakladatelství Karolinum, 2002. ISBN 80-246-0084-6.

[15] Univerzitní informační systém MENDELU. Základní pojmy teorie grafů. [Online] Mendelova univerzita v Brně. [cit. 2018-04-24] Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=9295.

[16] Univerzitní informační systém MENDELU. Reprezentace grafů. [Online] Mendelova univerzita v Brně. [cit. 2018-04-24] Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=9312.

[17] Grafy a grafové algoritmy. Distanční a prezenční vzdělávání informatiků. [Online] [cit. 2018-04-24] Dostupné z: https://phoenix.inf.upol.cz/esf/ucebni/Grafy_a_grafove_algoritmy.pdf.

[18] Reprezentace grafů. Mendelova univerzita v Brně. [Online] [cit. 2018-04-24] Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=19949.

[19] BARTUŠKA, Karel a Emanuel SVOBODA. Fyzika pro gymnázia: molekulová fyzika a termika. 4. přeprac. vyd. Praha: Prometheus, 2000, 244 s. ISBN 80-719 6200-7.

[20] Java – Největší česká online učebnice. ITnetworks. [Online] [cit. 2018-04-14] Dostupné z: https://www.itnetwork.cz/java.

[21] Java Essential Training. Tutorialspoint simply easy learning. [Online] [cit. 2018-04-12] Dostupné z: https://www.tutorialspoint.com/java_essential_training/index.asp.

[22] Aris Community [Online] [cit. 2018-05-10] Dostupné z: http://www.ariscommunity.com/aris-express

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

IDE Integrated Development Environment

TAČR Technologická agentura České republiky

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PI CD-ROM