KIV/ASWI

IMiGEr - DOT plugin

Bc. Vít Mazín
A18N0096P
mazinv@students.zcu.cz

20. 04. 2019

1 Formát DOT

DOT je jazyk popisující grafy. DOT grafy jsou typicky soubory s příponou *gv* nebo *dot*. Grafy popsané tímto jazykem tvoří jeden textový soubor, ve kterém je uložen celý popis grafu. DOT podporuje popis orientovaných i neorientovaných grafů a umožňuje vrcholům a hranám přidávat atributy. Protože je jazyk navržen obecně, lze jím popsat libovolný graf a přidávat objektům grafu libovolné atributy. Atributy bývají využívány pro grafický popis grafu, ale lze je využít pro uložení jakékoliv informace.

1.1 Neorientované grafy

Nejjednodušší objekt, který může být popsán DOT formátem je neorientovaný graf. Neorientovaný graf umožňuje zobrazit jednoduché vazby mezi objekty jako např. přátelství mezi lidmi. Klíčové slovo *graph* značí začátek neorientovaného grafu. Po tomto klíčovém slově následuje jméno grafu. Dále uvnitř složených závorek následuje popis vrcholů a hran mezi nimi. Hrany jsou zapsány jako dvě pomlčky mezi vrcholy. Takto je zapsán jednoduchý neorientovaný graf:

```
graph graphname {
    a -- b -- c;
    b -- d;
}
```

1.2 Orientované grafy

Podobně, jako neorientované grafy, lze popsat i orientované grafy. Orientovanými grafy lze popsat např. závislost balíčků programů. Způsob zápisu je velmi podobný zápisu neorientovaných grafů s tím rozdílem, že popis grafu začíná klíčovým slovem `digraph` a hrany mezi uzly jsou zapsány jako šipka (pomlčka a znak pro větší než). Tímto zápisem je popsán jednoduchý orientovaný graf:

```
digraph graphname {
    a -> b -> c;
    b -> d;
}
```

1.3 Orientované grafy

Atributy vrcholů a hran mohou nést různé informace o daném objektu. Atributy jsou zapisovány do hranatých závorek stylem klíč-hodnota. Příklad grafu s atributy:

```
graph graphname {
    a [label="Foo"];
    a -- b -- c [color=blue];
    b -- d [style=dotted];
}
```

2 Programová dokumentace

2.1 Architektura

Plugin pro převod DOT formátu na JSON formát programu IMiGEr je naprogramován jako modul programu. Modul je vytvořen dle šablony, kterou lze najít ve kořenové složce projektu jako *imiger-plugin-template*. Pokud je modul vytvořen dle této šablony a jedna zvolená třída tedy implementuje rozhraní `IModule`, je plugin automaticky zobrazen na hlavní stránce programu IMiGEr

2.2 DOT loader

Jedna ze tří hlavních komponent modulu je DOT loader. Byla vytvořena abstraktní třída `BaseDOTLoader` a následně její potomek `DigraphDOTLoader`. Abstraktní třída definuje, co má loader umět dělat. Loader ze zdrojového textu, ve kterém je zapsán graf ve formátu DOT musí umět vytvořit seznam vrcholů, hran a unikátních atributů. Pro naši implementaci loaderu `DigraphDOTLoader` byl použit parser DOT formátu *Digraph Parser*.

2.3 Graph factory

Druhá z hlavních komponent je graph factory. Byla vytvořena abstraktní třída `BaseGraphFactory` a její potomek `GraphFactory`. Abstraktní třída definuje především metodu pro načtení dat, která bere jako parametr DOT loader a metodu pro vrácení objektu `Graph`, který představuje vytvořený graf. Tento výsledný objekt je následně serializován do JSONu.

2.4 DOT

Tato třída implementuje rozhraní `IModule`. Metody z tohoto rozhraní vracejí název pluginu a výsledný JSON, který byl vytvořen převedením z DOT formátu.

3 Ukázka využití DOT formátu

Při odinstalování programu z OS Debian jsou často mazány i další balíčky. Abychom věděli, které balíčky mají být odstraněny a jaké mají společné závislosti, lze využít grafu v DOT formátu a jeho následné zobrazení v IMiGEr. Program *debtree* umí vytvořit graf ve formátu DOT, který obsahuje závislosti dotazovaného programu na ostatních balíčcích. Tento graf poté můžeme vizualizovat a upravovat v IMiGEr. Na obrázku 3.1 je vidět, na jakých balíčcích je závislý *Midnight Commander*. Díky možnosti vyjmout vrchol lze vrchol představující Midnight Commander odstranit a tím i zvýšit čitelnost grafu.

Obrázek 3.1: Závislosti programu Midnight Commander