Техническое задание, группа №2

20 февраля 2013 г.

Version 1.0

1 Введение

1.1 Назначение и область действия

Разрабатываетмый программный продукт предназначен для графического отображения графов, путём отображения содержимого .dot-файлов.

1.2 Краткий обзор

Разрабатываемый программный продукт позволяет загружать содержимое .dot-файлов, выполнять их графическое отображение, а так же позволяет выполнять произвольное редактирование графов с возможностью их последующего сохранения в формате .dot.

1.3 Назначение разработки

Выполнение командного задания по дисциплине “Проектирование Программного Обеспечения”

2 Общее описание

2.1 Функциональные харрактеристикам

Разрабатываемая модель должна обладать следующими функциями:

- Открывать файлы формата .dot и выполнять графическое отображение его содержимого
- Возможность произвольно менять вершины графа
- Делать масштабирование графа и навигацию по графу
- Выделение различных подграфов и связанных элементов
- Возможность сворачивания/разворачивания текста внутри вершин графа

2.2 Формат входного файла

Входной файл представляет собой файл с расширением *.dot* в котором информация о графах представлена в текстовом виде.
Для построения графа используется структура типа:

```
graph %имя_графа%
  {
    %описание_графа
  }
```

Для построения связей между графами используются символы “–” для неориентированного графа и “->” для ориентированного графа:

Неориентированный граф:

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

Ориентированный граф:

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

При описании графов можно использовать различные аттрибуты. В частности, можно менять тип фигуры вершины, можно добавить название, изменить цвет и тип как фигуры, так и линии.
Аттрибуты описываются парами *ключ=значение*, заключёнными в квадратные скобки. Та же язык DOT поддерживает задание нескольких аттрибутов. Аттрибуты разделяются пробелами.

Пример задания аттрибутов:

```
graph graphname
{
   // label - видимое название вершины
   a [label="Foo"];
   // shape - определение формы вершины
   b [shape=box];
   // color - определение цвета ребра
   a – b – c [color=blue];
   // style - определение стиля ребра
   b – d [style=dotted];
}
```

Пример задания множества аттрибутов:

```
digraph g
{
   node [shape=plaintext];
   A1 -> A2 [label="Same label" tailport=s headport=s];
}
```

2.3 Программная документация

Техническое задание составляется с использованием системы компьютерной вёрстки LaTeX. Программная документация составляется с использованием JavaDoc

3 Стадии и этапы разработки

Номер Этапа	Задание	Ответственный
Этап 1	1. Maven, класс хранения данных графа	Таммсаар Серафим
	2. Техническое задание и документация	Васильянов Георгий
	3. Парсер формата .dot	Бойцев Андрей