CS 207 Spring 2014 Final Project Documentation
by Aymen Jaffry, Joseph Pavlisko, Nikhil Sud

1) Work accomplished

SDL Viewer Extensions (by Aymen)
- node size, node color (shared with Conor/Soren)
- edge width, edge color
- window background

Animations (by Nikhil)
Fireworks

Graph Extensions (by Joe)
Random access node iterator

Force-directed Graph Drawing
Fruchterman Reingold (Nikhil)
Gauss-Seidel force layout- stabilize in concentric circles / stabilize in groups (Aymen)
Kamada Kawai with Simulated Annealing (Joe)

Data-gathering
Les Miserables (Aymen), Senate Races (Joe), Karate (Nikhil/Joe)

Other stuff
Documentation, readme, conceptual research (see PDFs in /research folder) (Nikhil)
Integration of code using templates into Project.cpp (Nikhil/Aymen)

2) Background Theory / Research

Some of the most flexible algorithms for calculating layouts of simple undirected graphs belong to a class known as force-directed algorithms. Also known as spring embedders, such algorithms calculate the layout of a graph using only information contained within the structure of the graph itself, rather than relying on domain-specific knowledge. Graphs drawn with these algorithms tend to be aesthetically pleasing, exhibit symmetries, and tend to produce crossing-free layouts for planar graphs - Stephen G. Kobourov, University of Arizona

Research papers on the individual algorithms can be found in the /research folder in the project home directory.

The algorithm of ***Fruchterman and Reingold*** relies on spring forces, similar to those in Hooke's law. There are repulsive forces between all nodes, but also attractive forces between nodes that are adjacent.

Alternatively, forces between the nodes can be computed based on their graph theoretic distances, determined by the lengths of shortest paths between them. The algorithm of ***Kamada and Kawai*** uses spring forces proportional to the graph theoretic distances. In general, force-directed methods define an objective function which maps each graph layout into a number in R+ representing the energy of the layout. This function is defined in such a way that low energies correspond to layouts in which adjacent nodes are near some pre-specified distance from each other, and in which non-adjacent nodes are well-spaced. A layout for a graph is then calculated by finding a (often local) minimum of this objective function