# Stryker MAKO

# Software Engineer C++ Assignment

**Specification document**

**Contents**

**Chapter 1: Introduction**

**Chapter 2: Features and Functionality**

**Chapter 3: Design**

**Chapter 4: Testing**

**Chapter 5: Future Enhancement**

**Chapter 6: Appendix**

**Chapter 7: References**

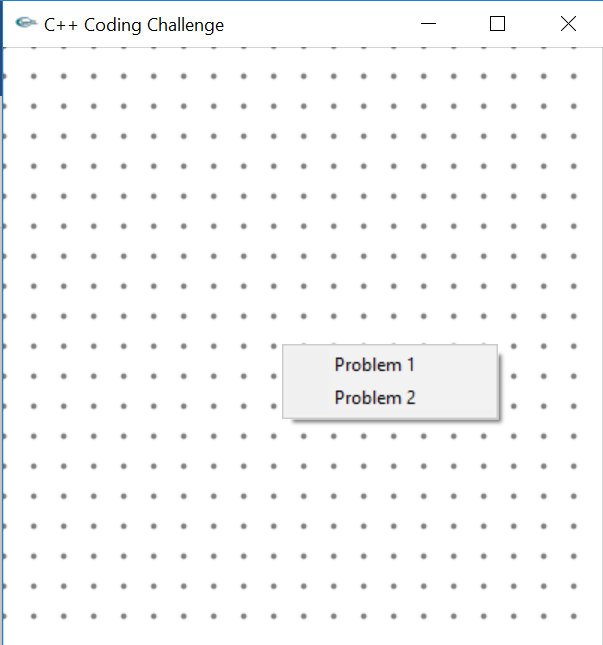
# Chapter 1: Introduction

This document serves as “Software Engineer C++ Assignment” specification document. The solution is developed in visual C++ 2017 using the provided template. It can be easily opened in visual C++ 2013. I have not changed the existing directory structure. I have not changed the design of provided template.

# Chapter 2: Features and Functionality

**How to Run?**

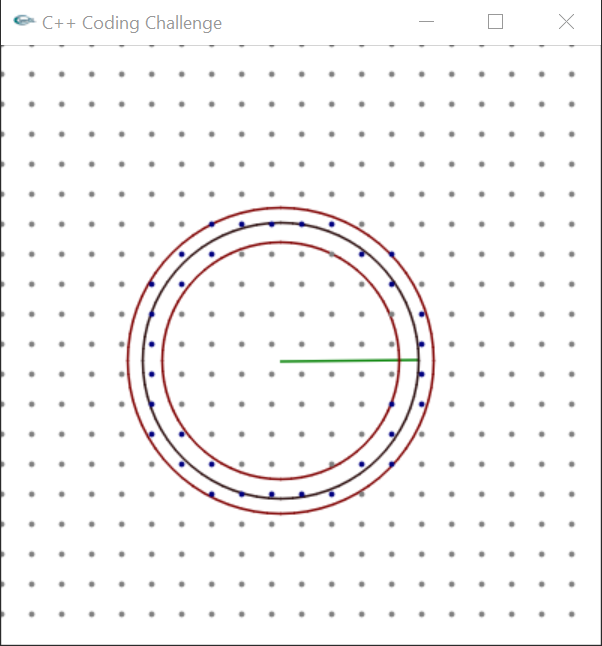
1. Take following files and place them to your desired location.
   1. Assignment\Release\Assignment.exe
   2. Assignment\Assignment\glut32.dll
2. Double click on Assignment.exe. It will open a grid(20X20)
3. Right click on grid to choose between problem1/problem2, as Shown Below



1. At any time on grid, press ‘c’ or ‘C’ to clear the grid whenever required.
2. Pressing ‘Esc’ will terminate the Program.

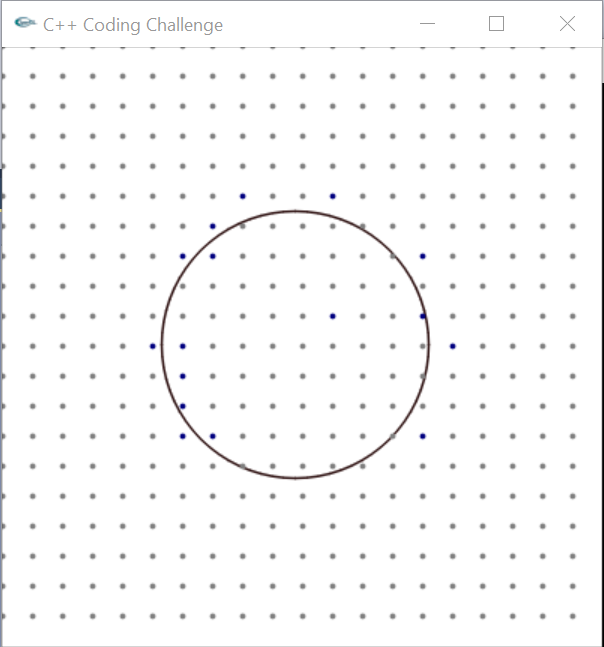
**Problem 1**

1. Left click and drag to draw a line. Once mouse left is Up it will draw three circles as shown Below.



**Problem 2**

1. Select points on grid
2. Press ‘g’ or ‘G’ on keyboard to create the best fit circle as Shown Below.



**Features:**

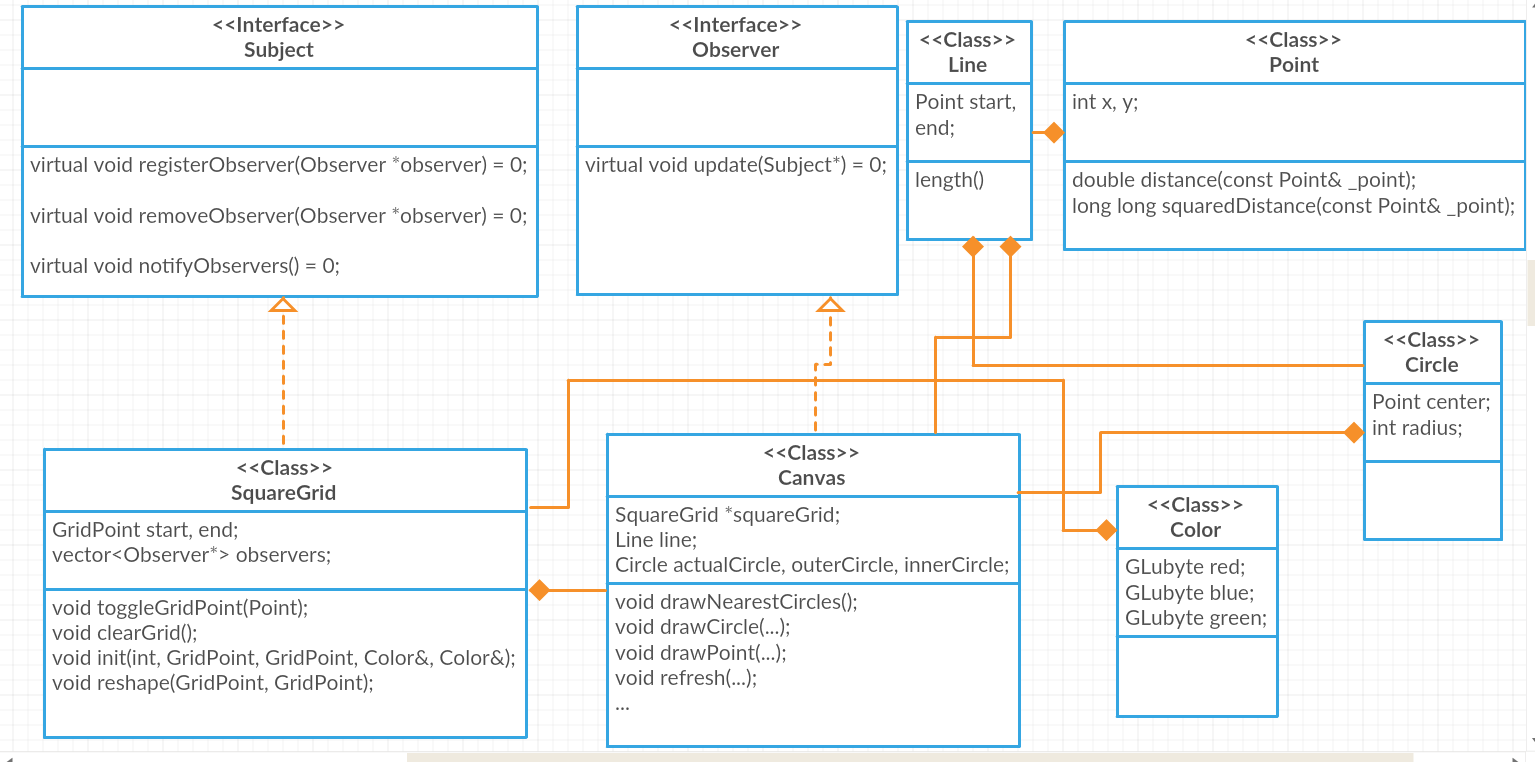
1. Color of Line, Circles is property of Canvas class and is editable from MyCanvas() constructor, GridPoints color is a property of SquareGrid, so that all GridPoints have same color at both state. It is editable from Canvas::initGrid() function.
2. Resizing of Canvas can be done at runtime, Grid gets reshaped accordingly. However other properties are reset.

# Chapter 3: Design

Design of This project is based on ***“Composite” and “Observer”*** design patterns.

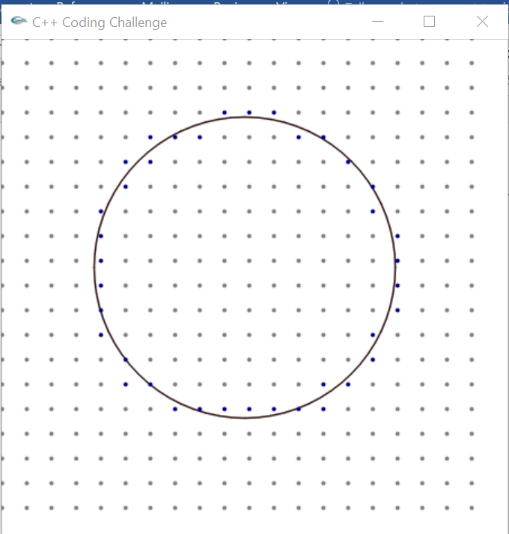
Currently there is only one Observer and one Subject. Grid is Subject and Canvas is Observer. Observer Pattern supports multiple observers and multiple publishers. Declared in SubjectAndObserver.h

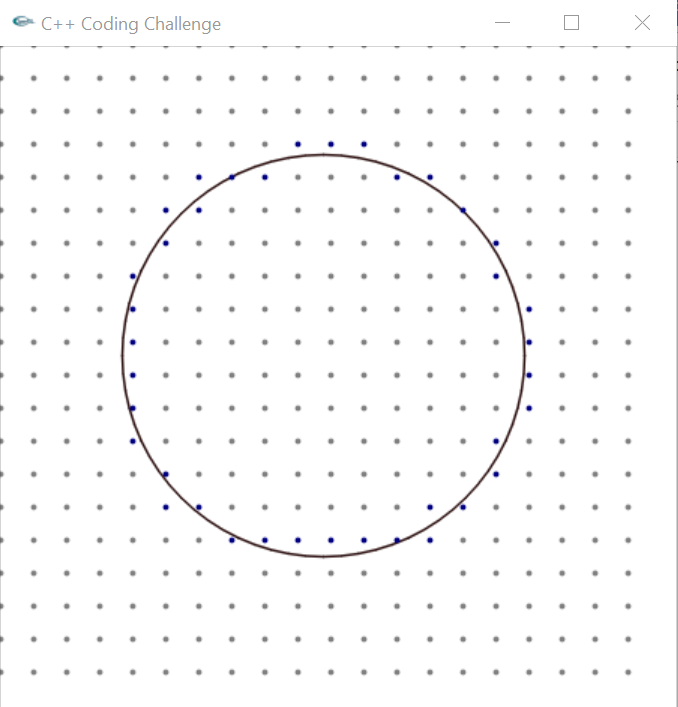
Below is the rough Class Diagram of project



# Chapter 3: Testing

I create the circle in problem1 and the points get highlighted. Then switch to problem 2 and press ‘g’

The Circle shift is very minimal as shown Below.



Very slight movement of circle towards ‘Y’ direction can be seen.

I tested manually at all corners of the grid and edges, to cover boundary test-case scenarios.

# Chapter 5: Future Enhancement

Following enhancements can be done.

1. Implement “Levenberg-Marquardt” from Reference #2 for dealing with outliers in Assignment\Assignment\GridUtil.cpp
2. Refactoring:
   1. Standardize Directory Structure
   2. Logging using log4cxx from <https://logging.apache.org/log4cxx/>
   3. Configuration (e.g. grid size, color) in either *filename.ini* file or database
3. Create Shape interface with pure virtual draw method. Every Shape class like Canvas, Line, Grid, Circle, Point, etc. should inherit Shape interface and implement draw Method.
4. Instead of clearing Objects on resizing of screen, they can be scaled.
5. Instead of Square Grid and Circle. Solution can be extended for Rectangular Grid and ellipse.

# Chapter 6: Appendix

**How to make the project compatible with VS 2013.**

Open Assignment.vcxproj file and change

<PlatformToolset>v141</PlatformToolset>

To

<PlatformToolset>v120</PlatformToolset>

Under

<PropertyGroup Condition="'$(Configuration)|$(Platform)'=='<***BUILD\_TYPE***>|Win32'" Label="Configuration">

Where

<***BUILD\_TYPE***> can be “Release” or “Debug” based on if you want Release binaries or Debug binaries respectively.

# Chapter 7: References

1. <https://www.opengl.org/resources/libraries/glut/>
2. <https://www.spaceroots.org/documents/circle/circle-fitting.pdf>