Creating MAZE and Finding Destination using various Graph algorithms

1. Introduction
2. Low Level Design
   1. Class Design and Architecture.
   2. Sequence Diagram.
3. How to Adding build Maze and Search algorithm.
4. IDE and Compiler

**1) Introduction**

In this application various Graph algorithms have been used to build Maze as well as find the path from source to destination on the Maze. Algorithms used as below.

Building Maze:

1. Depth First search.
2. Breath first search.

Finding or searching:

1. Depth First search
2. Breath first search.
3. Dijkstra Shortest path.
4. Bellman Ford.

Also it has the flexibility to enhance the application with new algorithm.

**2) Low Level Design: Class Diagrams:**

**Core Architecture:**



**Events:**



**Sequence Diagram:**

**Build Maze:**



**Find or Search Destination on Maze:**



1. How to Adding build Maze and Search algorithm.

Step 1: Add a new enum member into “enum eAlg\_Type” on file “cmazeconst.h”

Step 2: Inherit from class CGraphAlg and provide implementation based on the graph algorithms. And create this algorithm object in the function CGraphAlg::GetAlg. (Please refer to the existing algorithm implementation).

Step 3: Add a new algorithm name into Drop Down box on the GUI.

1. IDE and Compiler

**IDE: Code**::Blocks 13.12 (Window 32 bit)

**Compiler :** GNU GCC Compiler

**GUI Builder:** WxSmith (wxSmith is a free Rapid Application Design plug-in for [Code::Blocks](http://www.codeblocks.org/), a free IDE for C++, wxWidgets and other languages. wxSmith is designed to allows easy visual creation of wxWidgets window frames and dialogues within the IDE.