[FINAL PROJECT]  
MINESWEEPER

June 6, 2017

### This is an overview report. All detailed processes/reports are written on Trello - <https://trello.com/b/PTSfLNqW/minesweeper>

# Overview

## Project Plan Implementation

* **Find some application/website to manage the project processes : Microsoft Project , Microsoft OneNote , Trello => Choose** [**Trello**](https://trello.com) **because of the user-friendly interface.**
* **Base on the original Minesweeper game which play by mouse-clicking , start to research how to handle the mouse click at first , then learn to draw the graphic and control the console.**
* **Prepare some image in game (see** [**Section 5**](#_Images/Stocks_Preparation)**)**
* **Start to code level 1 to 4 since each level only need to modify/add a little of code : finish in one month.**
* **Find a framework to make the GUI of the game (level 5) : found out 3 potential framework (see Table 1) => choose SFML Framework due to its advantages.**
* **Start to learn the SFML Framework and design the UI : 2 weeks**
* **Continue to learn and code level 5 at the same time : 2 weeks**
* **Edit, comment, backup and package code.**

|  |  |  |
| --- | --- | --- |
|  | ADVANTAGES | DISADVANTAGES |
| **QT Framework** | **+ One of the biggest frameworks of C/C++**  **+ Support Drag-and-Drop UI Designing**  **+ Cross-platform building**  **+ Has its own IDE** | **+ Code involing JSON if using Drag’nDrop UI.**  **+ Lots of object-oriented code**  **+ Not fully compatible with Visual Studio** |
| **Cocos2d-x Framework** | **+ Best 2D game framework of C++**  **+ Cross-platform building**  **+ Wide support-community , free documents, example source codes.**  **+ Has its own IDE** | **+ Complex to install and use with Visual Studio**  **+ Object-oriented coding** |
| **SFML Framework** | **+ Easy to install and code**  **+ Wide support-community , free documents, example source codes.**  **+Not require too much object-oriented code (of course you can use it)** | **+ Doesn’t have its own IDE**  **+ The framework must go with Visual Studio to compile on another machine** |

**(Table 1 . Frameworks comparison)**

## Pseudo-code and Algorithm

* **First, get the mine field data from file or randomize the field base on width & height.**
* **Draw the game guide , game board .**
* **Set initial value for the counters (time , mines left , tiles opened…)**
* **Initial the hook to wait for first click on the game play area of user to start the game**
* **Get the position of each click and handle it base on click-type (left/right)**
* **If user open a tiles, check the data content :**
  + **Mine : Lose the game . Reveal the board and wait for user choice (restart/exit).**
  + **Number : Just show it.**
  + **Empty : Reveal the adjacent empty tiles.**
* **If all empty/number tiles opened , then user win the game.**

## Supported Techniques

+ Win32 API :  **allows user-written programs to interact with Windows, for example to display things on screen and get input from mouse and keyboard.**

* **Used functions :**
* **Hook mouse click**
* **Recognize the position clicked**
* **Control the console window :**
  + **Disable user-resizing actions**
  + **Resize console**
  + **Set color on console**
  + **Set the cursor position**
* **Draw the BMP images**

+ SFML Framework : **An object-oriented framework which provides a simple interface to the various components of your PC, to ease the development of games and multimedia applications. It is composed of five modules: system, window, graphics, audio and network.**

## Coding Structures

* **LevelX :**
  + **Header :** 
    - **Data.h : contains data structures and prototype of functions which handle the gameplay data such as mines generator , game board data, user clicking ,…**
    - **Graphic.h : contains prototype of functions which involve graphic such as drawing , resizing window, set the color …**
  + **Source :**
    - **Data.cpp : Implement the functions which declare in Data.h**
    - **Graphic.cpp : Implement the functions which declare in Graphic.h**
    - **Main.cpp : Main source file , initial variables , control game loop**

**(The same coding structure was also used for level 5 on the SFML Framework)**

## Images/Stocks Preparation

* **Some images was found on Google Image Search : Tiles blocks, level button..(**[**http://imgur.com/a/kdTHo**](http://imgur.com/a/kdTHo)**)**
* **Everything else (Main menu, buttons, color…) was designed by myself with Photoshop . The PTS files are placed in UI Design folder.**
* **Color mixing :** [**http://colorsupplyyy.com/app/**](http://colorsupplyyy.com/app/)

## Notes

* **Improvement :** 
  + **Displaying some image when user win/lose.**
  + **Designing a new style UI in level 5.**
* **Remaining Bugs/Errors :**
  + **The image displayed on losing game sometimes loses some pixels. (Haven’t found the reason)**
  + **The timer-counter on level 4 doesn’t auto-update in realtime. (Haven’t research the multi-threading in C++ due to overtime)**
  + **The first click can be a mine tiles (not the same as original game which the first click is surely a number/empty tiles).**

# Level of work accomplishment

|  |  |  |
| --- | --- | --- |
| Level | Percent of completion | Notes |
| **1** | **90%** | **Sometimes the “Lose” image lost some pixels while displaying** |
| **2** | **90%** | **Same as above** |
| **3** | **90%** | **Same as above** |
| **4** | **80%** | **Same as above. The time-counter only update when user move/click the mouse.** |
| **5** | **80%** | **Haven’t done the Highscore ranking function** |
| **6** | **0%** | **Haven’t done since time’s overdue** |