G52CPP, 2017/18, Coursework Part 3, GUI Framework

Overview

Coursework 3 aims to introduce you to a C++ framework for building GUI programs. It is a simplified cut-down framework covering just the essentials and aims to simplify the process for you. You will need to understand some C++ and OO concepts to be able to complete the coursework, but will not need to understand many complex features, so that it can be released earlier in the semester.

Getting started:

1. Download the zip file of the coursework framework. Unzip it and open it in Visual Studio. (See tutorial on getting started.) Compile and run the initial framework to ensure that it compiles and runs with no problems on your computer. Read the getting started document if you are stuck.
2. Do the framework exercise A and ensure that you understand about the basic features of the BaseEngine class and drawing the background.
3. Do the framework exercise B and ensure that you understand about the basic features of simple displayable objects.
4. Experiment with the framework. Try changing which object type is created in the mainfunction.cpp file (change which line is commented out) and look at the BouncingBall example and the Demos 1 to 4. Each should teach you new things that you can do and give examples of how to do them.
5. Start on the requirements below, using what you learned from exercises A and B.

General requirements:

1. Create a program which runs without crashing or errors and meets the functional requirements listed below and on the following page. Look at the Hall of Fame page to see some examples of previous work and decide on a basic idea for your program. You don't need to implement that much for part 3, but part 4 will involve finishing it.
2. Your program features which are assessed must be different from all of the demos and from exercises A and B. i.e. do not copy demo code or the lab exercises code too closely.
3. The aim of this coursework is the investigate your ability to understand existing C++ code and to extend it using sub-classing, rather than to use some alternative libraries:
   - Your program MUST use the supplied framework in the way in which it is intended. i.e. the way in which exercises A and B use it to draw the background and moving objects.
   - You must not alter the supplied framework classes.
   - You should not use any external libraries other than standard C++ class libraries, libraries used by the framework code.
4. When you have completed the coursework, demo your work to a lab helper and have it marked. There are 8 functional requirements, worth 1% of the module mark each, for a total of 8% of module mark for this part of the coursework.
5. Zip up your completed project and submit it to part 3 coursework submission on Moodle.

Functional requirements: (1 mark each, marker will tick off all that you have done on their mark sheet):

1. **Create an appropriate sub-class of BaseEngine with an appropriate background**. Create an appropriate new sub-class of the base engine. Name your class using your capitalised username followed by the text Engine. e.g. if your username was psxabc then your class would be called PsxabcEngine.

2. **Draw an appropriate and interesting background:** ensure that you use at least one of the shape drawing functions to draw on the background and that the appearance is different from the demos. I.e. show your understanding of how to draw a background for your program. Be prepared to explain what you have done to the marker if asked. A blank background will not get this mark – even if you change the colour – you MUST use one of the shape drawing functions (i.e. a DrawBackground…() function other than DrawBackgroundPixel()) to show your understanding.
3. **Create your own subclass of TileManager**. Create a subclass of the tile manager which has different behaviour (at least a little) to the demos. Name your class using your capitalised username followed by the text TileManager. e.g. if your username was psxabc then your class would be called PsxabcTileManager. Be prepared to explain the difference(s) from the demo versions to the marker. Hint: Look at the bouncing ball demo and demos 1 to 4. Display the tile manager on the background, so that the tiles are visible. (You can use it as a static background if you wish – it does not have to change in coursework part 3.) It must be different from the demos but can still be simple. You are just showing that you understand how to use the tile manager class. Be prepared to explain how this works to the marker if asked.
4. **Provide a user controlled moving object which is a sub-class of DisplayableObject:** Have a moving object that the user can move around, using either the keyboard or the mouse. i.e. show that you understand how to use the keyboard or mouse input functions. Be prepared to explain how this works to the marker if asked. Addition: due to requests for this, you may meet this requirement by having an object controlled by a mouse as long as you also have somewhere in your program where you demonstrate that you can handle keyboard input, and are able to answer questions about it during marking.
5. **Provide an automated moving object which is a sub-class of DisplayableObject:** Have another (separate to the user-controlled one) moving object whose movement is not directly controlled by the player, and which acts differently to the objects in the samples that I provided. i.e. show that you understand something about how to make an object move on its own. Be prepared to explain how this works to the marker if asked.
6. **Appearance of moving objects:** Give the moving objects an appearance which is different from those in the demos and different from each other. i.e. show that you understand how to draw objects. Ensure that when your objects move around, the screen is re-drawn correctly (i.e. no parts of objects left behind). Be prepared to explain what you did and how it differs from the demos to the marker if asked.
7. **Draw some text on the background.** Draw some text onto the background (not foreground) and ensure that the text is not obliterated when moving objects move over it. Ensure that moving objects can/do move over at least part of this text, to demonstrate that it returns after the object has moved. i.e. show that you understand how the background and foreground are drawn to and when to draw text. Be prepared to explain how this works to the marker if asked.
8. **Have some changing text, refreshing/redrawing appropriately which is drawn to the foreground (not background).** It may change over time (e.g. showing the current time, or a counter) or could show a score, for example. When it changes the old text should be appropriately removed. Be prepared to explain how this works to the marker if asked. This shows your understanding of drawing text to the foreground.