GAME1017 – Game Fundamentals II

Assignment 1

***** Due June 22st by 11:59pm *****

Assignment: 'Completing' Top-Down Shooter Game

You can work in pairs or individually.

The goal of this assignment is to expand a simple 2D shooter to include a singleton, state machine and UI buttons to change states. This assignment, worth 20% total, won't be broken down into different milestones but you'll have several headings going over each criterion due.

Mandatory Criteria

In this assignment, some criteria are marked as **Mandatory** which means each criterion marked as such must be completed, or you will receive a mark of 0 for the **entire** assignment. This is to ensure that students learn the most important aspects of the assignment.

Source Project

You will be provided with a starter project in which can you use to make the needed changes.

Requirements

Here's a summary of the new requirements for this assignment. You'll need to:

- Make the game engine a singleton.
- Add a state machine to run the game with:
  - Title state, game state, pause state, lose state, win state
- Add a UI button to the states for transitioning to the next.

Engine Singleton

As in the state machine example that we saw, you are to take the Game object and instead turn the Game class into a singleton thereby creating a globally-accessible instance you can access from any method of any class.

State Machine - Mandatory

I'll briefly go over each state and the requirements in point-form for each. If I don't mention anything, then please assume that you can have freedom to make your own decision for it.

A Title state will:

- Be the initial state.
- Display an image for the name of the game with size and name of your choice. It can just be a bitmap made in Paint. Use your judgement to determine the size on screen.
- Contain a small button with the text 'Start Game' on it. You don't need a mouseover state, but when you click the mouse when the cursor is over the button, the game will start.

The Game state will:

- Run the main game.
- See the headings below for additions to the gameplay.
- Parse a key press for 'P' that will trigger a transition to the Pause state.
- Still be rendered, but not updated, when the Pause state is active.

The Pause state will:

- Not replace the game state but will run 'over' it.
- Display a transparent rectangle over the game state.
- Display a button with 'Resume' text on it. Again, you don't need a mouseover state, but when you click the mouse when the cursor is over the button, the game will resume.

The Lose state will:

- Be triggered when the player collides with any of the hazards of the game, including enemies and the enemy bullets.
- Display a button with 'Main Menu' text on it. Again, you don't need a mouseover state, but when you click the mouse when the cursor is over the button, the title state will be triggered.
- You can display anything else you want in this screen but keep it clean.

The Win state will:

- Be triggered when at a condition of your choice, e.g. if the player destroys a certain number of enemies.

- Display a button with ‘Main Menu’ text on it. Again, you don’t need a mouseover state, but when you click the mouse when the cursor is over the button, the title state will be triggered.
- You can display anything else you want in this screen but keep it clean.

You can implement the state machine however you like so long as you have an abstract State base class, the required derived subclasses, and the state machine class itself to manage the states.

UI Buttons

The few buttons required to change states can be made however you like, even with Paint. Of course, you’ll need to write the code to support whatever method you choose.

Sounds

Sounds are provided in the starting project but you can add any more that you like including different tracks for the different states.

Misc. Notes

- You can work on this in pairs. Both partners MUST submit the same compressed project separately.
    - I should not have to say this, but if you work on it in pairs, both partners must do equal work. If one partner does not do anything, karma will ensure they fail at life.

Git/GitHub Requirement (0 marks but mandatory – see penalties below)

You will be required to create a Git repo for this and use GitHub. Git will be required for both assignments. You must provide the instructor access and submit your GitHub link with your assignment on Blackboard.

A1 Marks: 20% of course grade

Task	Marks	Description
Game/ State Machine*	10	You have a Game class as the engine, and it executes properly - 1 You create a static method 'getter' for the instance - 1 You create a static Game object and return it from the method – 1 You access the static instance method in your program – 1 You have an abstract State base class – 1 You have a subclass for the title state that works properly – 1 You have a subclass for the game state that works properly – 1 You have a subclass for the pause state that works properly – 1 You have a subclass for the lose (and win) states that work properly – 1 You have a state machine class that controls the states and includes methods for update, render, enter and exit and changing states – 1
UI Buttons	5	You created the button for the title state properly – 1 You created the button for the pause state properly – 1 You created the buttons for the lose (and win) state properly – 1 The buttons are rendered over their state – 1 The player can properly click on the buttons and trigger the state change – 1
Aesthetics	5	Your game compiles and executes properly - 1 Your game looks clean and plays smoothly – 1 The transitions between states work and are smooth – 1 There is no delay in playing sounds when they are triggered - 1 There is no flickering when the pause state is running - 1
Total:	**20**	*This criterion is mandatory. If you don't have it, you get 0.

Assignment 1 Submission:

- Before you zip, make sure to show hidden files/directories first and delete the entire .vs folder if applicable. Zip your entire project and submit it through the Blackboard link as I want to make sure you have got the proper include/library/other links in your project settings.
- You must provide the instructor access and submit your GitHub link with your assignment on Blackboard
- **You must submit a video showcasing your game and explaining your framework improvements**

Name it: **GAME1017_A1_LastnameFirstname.zip**
or if working in pairs: **GAME1017_A1_Lastname1Lastname2.zip**

Penalties:

- No GitHub link submission or invalid link/repo: 0% for submission
- You submit only a .sln file: 0% for submission
- If I cannot play your game, i.e. it won't compile: 0% for submission
- You code it in any language other than C++ with SDL: 0% for submission
- **No explanation video: 0% for submission**
- You have a .vs path (bloat or not) in your submission: 50% off submission
- Wrong naming convention: 50% off submission
  - So, make sure you take 10 seconds and rename it!
- Late penalty: 10% per day late up to 50%, then not accepted at all