Lebanese American University
School of Arts and Sciences
Department of Computer Science and Mathematics

CSC458– Game Programming - Project 2 (a group project)

Date assigned: Monday 4th of April 2022 – 11:00 AM
Date Due: Monday 25th of April 2022 at 11:45 PM.

Objectives:

To test students' knowledge of the following Unity topics:

Movement and rotation in 2D or 3D	Scriptable Objects
Collisions and Triggers.	Saving game data to files (persistence)
UI in Unity	Animations, Animation controllers, Animation events
Camera follow/Cinemachine/Cutscenes	Scene management & Levels
Serialization of variables & classes.	Coroutines
SFX & Music, Singleton and other design patterns	Raycasting
User Input from the Keyboard and the Mouse	Navigation meshes, NPCs, AI
Particle Systems	Events and delegates
Object Pooling	Post processing, TDD and GDD

Genre and topic/theme

This project gives the students the freedom to choose the genre/sub-genre except few genres that you are not allowed to choose. Please continue reading this document so that know what genre/themes you are NOT allowed to choose from.

The game submitted could be a 2D game with perspectives either "Top down", "Side-on" or "Isometric 2.5D", kindly check this Unity manual link for your 2D game options or it can be a 3D game **with 3rd person perspective ONLY**. No first-person perspective is allowed at all. In 2D, you are forbidden to create an endless runner or any of its flavors or subcategories.

Concerning the theme instead of giving you one theme like game jams, I have decided to give you more than one theme. Precisely, you have four themes to choose from:

- First theme which you can choose: "**Dualism**".
- Second theme which you can choose: "**Everything falls apart**".
- Third theme which you can choose: "**knights, blades and honor**"
- Fourth theme which you can choose: "**Spooky**".

The theme will open new ideas, elements, and gameplay mechanics. The theme will govern literally the assets chosen in your game world. Be creative and enjoy working on the project! This is an opportunity to learn!

Please fill in ASAP the data required in **the shared Excel sheet**. You need to reserve a group number even if you are doing the project alone. You must include in the Excel file: the full names of all your team members in addition

to the type of game chosen (2D vs 3D), the genre/sub-genre chosen and the theme which should be chosen only from the 4 detailed above.

General Rules

- Late submissions are not allowed. This is **a group assessment**. Maximum number of students forming a group is 4. You can do the project alone or with 2 or 3 students if you like. In all cases, whether alone or with other students, you must fill in your full name and the full names of your team group members in the Excel sheet shared and you must reserve a group number.
- **Cheating or copying** other students' work will get all the involved students a zero grade. I do not care if someone copied your work or who copied from whom. The project assignment should be unique. It is statistically impossible that two groups in the class would end up doing the same project or have the same or even similar code.
- **Any form of plagiarism or academic misconduct is prohibited. I have a zero-tolerance policy concerning this**. Copying code found elsewhere or changing it a bit and then claiming it to be yours will get you a zero grade immediately. **Using anything which you did not create yourself, MUST be cited and credited adequately** in a separate MS Document which you include in your submitted Unity project zip file. Using any game asset, any form of code (small or big in size), pseudocode, ideas, techniques, from any public or private source, online or in any other form, from a video on YouTube or from a Udemy/Coursera/Skillshare course or similar platforms or from a book or from an article etc… without citing the exact source adequately in your accompanying report **will get a zero grade on project 2 for all the group members.** This is literally the definition of plagiarism.
- The **Unity code/project submitted should not appear elsewhere**, **nor should be a previous submission to a different or to the same course in any previous semester or in any other university or academic institution,** nor should be a project made elsewhere in the industry, nor made for a client, nor made for a game jam, nor made before for any other reasons that I did not state. If we discover this is the case, you will get a zero grade immediately and an expulsion from the entire course on first offence.
- Your game **must NOT contain any content (small or big) that include anything NSFW (Not Safe for Work) or offensive/discriminatory/inappropriate**. In addition, **your content must NOT be about politics (foreign or especially local) NOR about religions or matters of faith and belief**. I respect freedom of speech, but game programming is not the venue to express it. In addition, I do not want the project to become a political propaganda. Any failure to adhere to any of these rules will get you a zero grade even if you submit the best game in the world.
- Please refer to the student code of conduct in the syllabus for further penalties on academic offences.

What deliverables I need to submit?

Each group member MUST submit the following: (please include the group number in all the MS Word documents):

1. **Citations and Contributions (as an MS Word document)**: In this document, you must include all the references of **anything & everything** that is not your own creation. There **must** be a contributions section in this document, which **should be written by each group member**. In this section, you must state exactly what is your contribution to the project in details and that of each group member. Make sure that the contributions are **relatively equal** since this will negatively affect your grade if you do not contribute much.
2. A **Game Design Document (as MS Word Document)**: must be submitted. It is a document detailing the game mechanics, levels, genre, theme, and other topics discussed in class. It should not be more than 2 pages.

3. A **Technical Design Document (as MS Word Document)** containing flowcharts or sequence diagrams and important class diagrams with explanation of the most pertinent code.
4. **A Unity project with all the files & the folders except the "Library" and "UserSettings" folders**. Please only delete these 2 folders after you completely finish your Unity game. The Unity engine should be closed when you delete these folders. Why? these folders i.e. "Library" and "UserSettings are auto generated by the Unity Editor when you open the project. This is to make sure that the folder size of your Unity project remains small. Always make sure that you backup your game on a regular basis. You can use a version control system or Unity collaborate. Please make sure that project 2 repository is always private/hidden and no one can see it before submission.

Put all the above in one single folder and create a zip file with **your full name the way it appears on LAU Banner** (FirstName-LastName.zip) and upload it to LAU blackboard. If you are doing the project alone: you still need to submit all the documents & the Unity project files as detailed in the points above but there no need of course to include the contributions section in the Citations MS Word Document since you are doing the project alone. In this case, just include the citations in the citations MS Word document.

NB: If project 2 final file size is still large (after you have removed the Library and UserSettings folders) & you cannot upload it to BB, please upload it to OneDrive or Google Drive or similar cloud services and share the link only with the lecturer and/or with the TA before the deadline. The link should remain active and accessible till the end of the course with no changes in the files nor in their metadata. If we discover any change even a minor change, all the group members will have zeros. Any change in the project after submission is a serious academic offence and will have disciplinary actions especially if done to fool the lecturer or the TAs in the demos and especially when the lecturer and the TA were not notified of the changes made. Don't leave submission till the last few hours. We live in an unstable country with unstable internet and electricity so ideally you should finish early if you have good time management. You have a lot of time to finish this project.

All group members MUST demo their project in front of the lecturer or the TA or both otherwise you will get a zero grade as I have to check that it is actually your own work not someone else work. If you do not understand your code or some feature in your game, this means that you did not write it, or this could mean that you did not do the project or part of it. Don't let me think this way. You can learn to do things from the web. Nobody reinvents the wheel but do not use something in your project that you do not understand. There are no pedagogical benefits in doing this anyway. If you do this, it will negatively affect your grade. **Penalty of not understanding code could reach deduction of 45% of the total grade**. Bear in mind that a true creator always knows well his/her own creation.

The project is an opportunity for you to learn new things so please impress me! Make sure you do a good project so you can impress prospective employers and so that you can put it in your GitHub/bitbucket profile or portfolio.

Please read the requirements slowly and carefully. **Your project 2 MUST adhere to the following requirements to get the full grade – each point is assigned a certain weight of the total grade**. **NB:** I did not specify dimensions nor quantities concerning game features and things like that. Please use whatever you find convenient. Please pay attention and make sure you do not commit academic misconduct or plagiarism. No need to plagiarize and thus take zero. You must cite everything.

Common Technical Requirements (for both 2D or 3D games) – apply to all

1. **Movement/rotation requirement**: For moving, rotating, jumping, crouching etc… your main player, you can use a ready-made character controller/movement script(s) or create your own logic using any movement/rotating method (rigidbody, transform, character controller, Navigation Mesh Agent technique etc..). If you use code from online sources, you must understand how the code works otherwise you will be *severely* penalized during the demo. Of course, you must cite the source of whatever you take from

others adequately otherwise you will take a zero grade. If you decide to do a 3D game and 3rd person perspective, you can use if you like the Starter Assets - Third Person Character Controller from Unity Technologies. Input should only be via Keyboard and mouse. The game builds should be for MS Windows or MacOS only.

2. **Levels and Level Design requirement**: You must have **at least 2 levels** that are complete and meaningful according to your genre and according to the theme. I need to see scene management functionality from C# code. Your levels do not have to be massive levels. Design them well and keep things simple and beautiful. Simplicity is beauty! Make sure that you always create an MVP game first then expand later if you have time. You can use anything you want to create your game world. **Example:** terrain tool, or any other platform. Be creative!
3. **Content of each level requirement**: your game must contain buildings, monuments, environment features (Rocky Mountains or forests or medieval castles etc…) that are pertinent to your chosen theme. Models and other assets can be acquired from free resources online as long as you mention their sources. Make sure you choose assets that are compatible with the render pipeline you are working with. Each game level should feature several elements depending on the theme and genre chosen such as ammo pickups, powerups, obstacles, collectibles, and others. Esthetically pleasing game content will get a higher grade.
4. **NavMesh/Enemies/NPCs requirement**: You need to create enemies of some sort depending on the theme that you have choosen. Enemies must decrease your health when they attack you. I am giving you the freedom to decide scenarios of what happens when you die or when you win. For enemies or other Non-Player Characters (NPCs) you must use the NavMesh technique. I need to see *some clever behavior* in enemies and/or NPCs such as intelligent patrolling, intelligent attacks perhaps when in range or something better than this. It is good to see that you can simulate enemies' behaviors such being suspicious or dwelling in a certain patrol waypoint for a while and then move on or you can add some random behavior to give the impression that the enemies and/or NPCs are acting intelligently. Now to get the maximum/top grade of this requirement: you can either (1) code an FSM (Finite State Machine) to represent transitions between states of NPCs and/or enemies. OR (2) you can also use behavior trees or use both methods (even better). To get you started with FSM consult these links: (Link1, Link2, Link3) and for Behavior Trees consult this (Link1) – you can find tons of videos on YouTube and on famous educational MOOCs and these topics are covered in lectures. Please pay attention to plagiarism!
5. **Camera follow/Cinemachine requirement**: the camera must follow the player in a smooth and efficient way. You are advised to use the Cinemachine package, but you can use any other method. I will not force a choice here.
6. **UI requirement**: you must have full intuitive meaningful UI. Esthetically pleasing UI will get a higher grade on this requirement. This requirement will flex further your UI muscles, so it is a great opportunity for you to learn new techniques. I need to see:
    - **A Main UI Menu**: that greets the player at the start and asks the player whether to play, or to load a saved play (more on this in the requirement dedicated to the saving system), a button for the instructions that opens up another UI menu that explains how to play, another bottom for quitting the game, a button for the settings of the game where you can change settings such as changing the volume of the background music and/or changing the resolution of the screen and/or anything else really as long as it is a useful and meaningful setting for your game. When the quit button is pressed you need to ask the user if they are sure of quitting and provide them with the necessary buttons. Quitting is just a matter of calling Application.Quit() to quit the game. If their play is not saved, you need to ask them to save and provide them with a save intuitive UI that let them save the game. Please check the saving requirement for what to actually use when saving and loading previously saved gameplay.

- **A Settings UI Menu**: As was mentioned in Main UI menu paragraph there exist a Settings UI menu. You need to create a settings menu containing your settings for the game. To take a full grade on this requirement you must include at least one setting to change by the player such as the volume, or resolution or anything else. I would like you to consult Brackeys YouTube video on the topic found here.
- **Instructions/Tutorial UI**: as detailed in the main UI menu, the instructions UI can be designed as you wish. This UI must give the player all the knowledge necessary to play the game.
- **A Save UI Menu**: This UI should be simple and would ask the user to write a save filename. Example you can have an input field that takes the name of the save and create the file or do this automatically without bothering with an input field. There should be at minimum two buttons Save button and back button. Now you have the freedom here to add anything you need or find convenient to add.
- **An in-game UI**: You should have depending on your genre/sub-genre some beautiful unobtrusive in-game UI such small labels/ or other stuff to show things such as score, pickups, powerups, Mana (magic), ammo, inventory UI etc… This is of course depending on your genre and theme. I am giving you the freedom here but I need to see in-game UI to get the grade of this part of the requirement.
- **A Pause/Resume UI**: During the gameplay, the player must have the ability to press a button on the keyboard (of your choosing) to pause the game and then another to resume the game. When you pause the game, you must show a UI menu that include the ability to leave the game or to change the settings (you can link to the settings menu UI here) or to resume. You must also have in-game UI for the functionality of resuming/pausing like maybe two unobtrusive small, beautiful buttons that allow the player at any time during the game to pause and resume the game via mouse. Consider using Time.Timescale for the effect of pausing and resuming the game.

7. **Background Music and singleton requirement**: You must use a convenient background music (not too loud nor should it be annoying) that could last for all your levels or at least it must last more than 1 level and it should not restart on each level. Choose it based on the theme. You can have voice over if you like in certain parts of the game to add some panache! Consider using the singleton pattern and DontDestroyOnLoad technique which you have learned in one of the lectures.
8. **SFXs requirement**: I need to see a good amount of SFX (sound effects) in your game. SFX should be used in a meaningful way, and you should show that you are manipulating these effects from the C# code.
9. **Object pooling requirement**: Instantiating and destroying game objects are very handy and essential. However, we want to make sure that your game is performant. You must show me at least once that you have use an object pooling system in your game.
10. **Saving system requirement**: You need to have a complete functional saving system to disk in your game. You need to save the location/rotation of your player, the score, mana or any other important elements … in your game depending on your genre. The user should have the ability to load a saved game and resume her play from where the game was saved. You can use playerrefs for saving settings of you game such as the volume, resolution and other things which would load automatically when the game restarts but the main gameplay save should be on disk (i.e. via serialization).
11. **Raycasting requirement**: Raycasting should be used in your project at least once in a meaningful way to get the full grade of this requirement. Bear in mind raycasting is very useful in games. Raycasting could be used for shooting but not only that. It can be use to move to a location clicked on the world map (such as in RPG, strategy or similar games) or to check that there are obstacles between your player and the enemies or for a million other usage. Show me one usage in code and bob is your uncle!
12. **Coroutines requirement**: I need to see that you have used coroutines at least once in meaningful way to get the full grade of this requirement. You probably will use more than 1 coroutine. Coroutines are extremely useful. Recall from the lecture on Coroutines that they are used to do tasks that spread over

time or across different frames such as dimming slowly a light or fading slowly a color or moving an enemy slowly etc...

13. **Animations/Animator Controllers/Animation Events requirement**: your main character, all the enemies and/or NPCs must be animated in a meaningful way. I need to see in C# code that you are manipulating and triggering animations in a meaningful way. I need to see that you are using animation events.
14. **Particle effects requirement**: Your game should contain particle system(s) in a meaningful way. You have used this in project 1 so this should be easy. Please I need to see that you are controlling the particle system via C# code in order to get the full grade of this requirement.
15. **Collisions & triggers requirement**: your game must contain good amount of collision logic and triggers. Triggers can be used to set hidden game objects which when you pass though trigger something: turn on/off lights or set an explosion or sound an alarm. It is up to your imagination!
16. **Scriptable objects requirement**: I need to see you using scriptable objects in your game. Scriptable objects are super useful. They are data containers that store large quantities of shared data independent of class instances. They have many usages: they can populate UI such as inventories, or store information…
17. **Events and delegates requirement**: I need to see you use delegates and events in your project in a meaningful way. Resources to consult: Link1, Link2, Link3.
18. **Post processing**: Post processing is a process that adds image effects that can be applied to change the look of the game (final touches if you like). You must apply at least 1 nice image effect you like to take the full grade of post processing requirement. Pay attention to what render pipeline you choose. Resources: Link.
19. **Serialization & game tweaking in inspector requirement**: I need to see a good number of serialized variables so that your game can be tweaked easily and efficiently from the Unity Editor (via inspectors) by future game designers 😊. You can create ranges of values and several serialized variables of different types.
20. **Coding style/code tidiness requirement**: good coding style/usage of comments/DRY code/good overall program hygiene/Correct OOP. Kindly add tooltips to the serialized variables in the inspector to explain briefly what these variables mean so that the TA would understand what your serialized fields do in the Inspector. Add other Unity attributes such as Range or [Header("XYZ")] attribute appropriately etc…

3D Technical requirements specific (only for those who choose to do a 3D game)

1. For those teams that choose to do a 3D game, the 3rd person perspective is the only one that is allowed (example: Role Playing Games (RPG), Third-person shooter (TPS), strategy game etc…), no first-person perspective game is allowed in project 2.

2D Technical requirements (only for those who choose to do a 2D game)

1. For those teams who choose 2D, you have all genres/sub-genre except endless runner and all its flavors.

Bonus Grades (non-essential requirements)

NB: you can get 100 if you do all the essential requirements asked. The bonus grades are additional grades. They will help you raise your grade in case you did not manage to do an essential requirement or part of a requirement:

1. Show me in your project that you have used Gizmos drawing in C# code to show gizmos around important things in your game in the Unity Editor (not game view). Gizmos are helpful for game developers and designers. For the sake of giving few examples: you can create Gizmos in C# to delineate the chasing region of your enemies, or you can draw the patrol points & the paths between them etc…. All gizmo drawing must be done in either OnDrawGizmos or OnDrawGizmosSelected methods. These methods are

like the Unity methods Update, FixedUpdate, Start, LateUpdate etc... The Gizmos Unity API is extremely simple. You can find it here. [+3%]

2. Usage of any external software (such as Blender, Maya, Audacity, Photoshop etc…) to create your own assets: models or SFX, VFX or animations etc... will be rewarded in project 2 [+3%].
3. You will get bonus grades [+3%] if you use anything that is more advanced AI wise than what is stated in requirement "NavMesh/Enemies/NPCs requirement" in the Common Requirements (for both 2D or 3D games) [+5%].
4. It would be nice to see based on certain *moments* in your game, different cameras' views are shown (like rendering from another camera which is showing different part of your scene or showing enemies starting to engage based on an event or to show some cutscenes). Use the timeline tool in Unity if you like. Consult this video for a good starting point [+5%].
5. The correct and meaningful usage of any other game design pattern - on the condition that it is not used in any of the essential requirements - will guarantee you with a [+3%] bonus points. You can check a very good resource here.

Remember a project such as this aims to teach you things that might not be covered yet in class. Although we will try to cover the topics, we might not cover everything.

Good Luck!