**Milestone 4 Review:** Postmortem

**Name:** Jayson Fitch

**Team:** Four Guns

**Overview:**

Four Guns was a great project, and I really enjoyed working on it with the team. A lot of the content that we wanted to put into the game did not get finished by milestone 4 unfortunately, but we plan to finish the game over the summer. The two main causes for this lack of content were poor scheduling and time management, and over-scoping.

**Scope issues:**

Much of the original ideas we had were not focused on game mechanics but rather on story, cool interactive “mission hubs”, implementing different game modes, and a plethora of other very *specific* ideas that were unnecessary for simply building a game that works and is fun to play. We were all dreaming a bit large about what *Four Guns* could be, and unfortunately that backfired on us. Initial implementation of the engine to begin with was successful however.

**Timing/Scheduling issues:**

In the first few weeks Dan and Sean had worked out most of the engine and architecture, and Jim and I were busy working on artwork and music. We ended up having to divide some of Dan and Sean’s plans between Jim and me so that we would be able to get a grade for the programming sections, and Shiv was busy working on the level editor. This ended up working out for the most part because we got a fun multiplayer game after the second week with some really great looking artwork. I believe getting to this stage so quickly was a mistake however, because we let the rest of the project fall onto the backburner while we all got other homework done.

**What went right & Conclusion:**

For the most part things went right for the project. We have a working prototype that has some interesting elements and is fun to play. I enjoyed working with everyone, and this project really helped me to realize that I would much rather be an artist than a programmer (along with Professor Schwartz’s class). This project also gave me a great insight into what working in a game-design team is like and added a great piece to my portfolio which will both be invaluable to me in the coming years. I hope to continue to work with this team and that the project becomes much more than a prototype over the summer.

**Milestone 3 Review**

**Name:** Jayson Fitch

**Team:** Four Guns

What was your role for this milestone?

I took on the role of (an) Architect, in a way. I took on any odd jobs that needed to be done, and built any necessary content.

Which data structures did you develop? Why did you pick that data structure over another?

I developed/implemented the Stack data structure. I did this by using a stack to dictate game states--for instance when one pauses the game it pushes the "paused" state on top, and when one presses the "b/escape" button it pops the current state off the Stack and returns the user to the previous state. I chose this structure because it works exactly like stacking things on top of each other, which is what our menus do in a nutshell.

Space complexity: The stack stores data at Θ(1). It stores the data in a new node and moves the head pointer. If the number of items in the stack increases to above the maximum amount allowed, it calls RemoveAt which is still Θ(1) with respect to space because the method loops through all of the data and changes references, not actual locations of data. Peek and Pop are also Θ(1) because no additional data is stored.

Time complexity: The stack's push time is Θ(1). There are no loops and the method simply adds a node to the underlying list. Both peek and pop are also Θ(1) because peek simply returns the head node and pop points head to its next node and reduces the count. If, however, the maximum size is reached, the RemoveAt function is called which has a time of Θ(n) because it must loop through all of the nodes (where n is the number of nodes).

Explain the features you worked on for this milestone.

I worked on the StaticGameObj class, implementing the stacked menus feature, and added a running feature to the player actor.

StaticGameObj: We needed to figure out a way to define objects that players could collide with but were also perhaps texture tiles, or even buildings. I created a series of constructors to define collidable objects (along with the objects themselves in the content) to be used when loading a map. Shiv is implementing a dictionary for each collidable type (depending on if bullets/actors can pass through and the size of the rectangle) in order to make this more efficient.

Stacked Menus: I had to change the way game states were modified, from simply using the enums and a "gamestate" variable to using a stack of game states (Stack<GameState>) and added code to add, pop, and peek states when necessary. This was somewhat difficult due to a lack of commented code.

Running: I added the ability to run into the game. We had to decide if we wanted to limit the player’s abilities and allow them to run infinitely or only allow them to run for a short amount of time. We decided on limiting their ability to shoot while running in order to make the game faster and "more hectic.”

Does your code have any bugs or issues that need to be fixed?

We currently have an issue with collisions only working in the "Y" direction for collidable objects, and collision rectangles being a bit too large. The dictionary will have to be implemented before we can completely resolve this issue (effectively). We're also having an issue with the list class in which a null reference exception is being thrown. (I wonder if perhaps this could be fixed with a try/catch? I will have to bring this up in our next meeting).

Has the design, architecture, look, or timeline of the overall game changed since milestone 2?

Currently the only notable difference is in our timeline. We were planning to be pumping out content and be finished with code at this point, which should be the case as long as everyone pulls his weight for milestone 3.

**Milestone 2 Programming Review**

**Name:** Jayson Fitch

**Team:** Four Guns

What was your role for this milestone?

I took on the role of Architect, in a way. I took on any odd jobs that needed to be done, and built any necessary content.

Explain the features you worked on for this milestone.

Saving and Loading players (not integrated yet, technically): The PlayerSave class saves or loads a player in binary format, interacts with the Player class in order to get or set player data, and interacts with the Gun class in order to parse a string of weapon details for the player's inventory.

Problems to overcome: We had to decide what format to save in. Originally we wanted to save in XML but we don't want the user to be able to edit player files easily (in order to limit cheating). We also had to figure out how to save and load weapons, and for now we decided to parse the weapons and save them as a string of data.

Does your code have any bugs or issues that need to be fixed?

There are no current bugs, however saving and loading is not yet implemented into the game. It has been tested and is working though.

Has the design, architecture, look, or timeline of the overall game changed since milestone 1?

We decided to move from parsing XML to saving and loading binary files considering we have a level editor, and we may end up adding a gun editor to make sure options are limited for weapons (no 'super guns').