**Game Design and Development  Milestone 4 Write-Up: Postmortem**

**by: Mass Extinguisher**

**Robby Piwko - Design/Leader**

**Bailey Russo - Architecture**

**Aiden Thinn - Interface/External Tool**

**Design/Leader/ Architecture**

**Robby Piwko**

*What I did.*

I coded the entire level, the Gravity function, platform collisions / physics (vertical and horizontal), camera movement, made it so the player can win, respawning enemies when reaching the npc, resetting the level without closing console, person taking damage from fire and turning red, coded and implemented the NPC, Ground, and House classes, implemented the external tool in the game class, implemented most art assets into the game class, and shooting and jumping functions and Fire and Water classes with Bailey

*What went right?*

I think our concept went very right.  I really like the idea of the backtracking through the level to escape while having a handicap, and I think our idea of carrying someone back is a good way to do that.  I also really like the idea of a superhero firefighter.  In a full version of the game, I could see a lot of the ideas we have here expanded upon. (I explain a lot more about this in the ‘differently’ section.)

Coding the level definitely went right.  Once we had the house background, I was able to easily hard code where all the ground and fire objects went.

The objective was also fairly easy to code, doing it by checking if the player intersects with different arbitrary rectangles.

*What went wrong?*

The platform collision, gravity, and jumping.  The gravity was difficult to figure out.  I tried doing it on my own, but ultimately went to the Prof for help.  I had right idea, but wasn’t executing it correctly so he showed me how to modify the code I had in order to make it work.  The jumping was a similar situation.  I had it working, but only with one platform and it was very stuttery.  Once I asked the Prof, he helped me optimize it so it wouldn’t stutter and showed me how to get it working with multiple platforms.  It was stuttering because I was doing too many loops.

Shooting and fire detection.  Getting the player to shoot was very hard.  The ‘first draft’ would reset the position if you shot it more than once, it would only work if interested perfectly with the fire, and would spawn at (0,0).  Bailey and I had a hard time figuring out to get everything to work.  It took a few weeks, and help from a few TAs to get it working right.

The animations.  One of the last things we tried to do was animate the player walking.  Bailey tried to figure it out, but was only able to get him turning around.  We all tried to figure it out, but no luck.  It’s not a big deal, but could have been a nice extra touch.

Also, a team member dropped out, so we all had to do more work.  This threw us all off.  Stanley didn’t do anything before he left, so we had to make up everything he didn’t do.

*What did you learn?*

Stuff takes a long time.  Because Stanley dropped out, we had to cut a few things.  As time went on, we realized how much time everything took, so we had to keep cutting stuff.

You have to do the basic mechanics before anything else, and they take a long time to perfect.  If I would’ve know how much time getting the basic mechanics to work would take, I would have put in a lot more time upfront, rather than taking my time.  Because they took so long to complete, we had to cut a lot more than expected.

*What would you do differently?*

I don’t think I would’ve done too much different concept wise.  I think we were overambitious, but I think if we had the whole year to finish the game, we could have included the ax and multiple levels, and more.  If this were to become a bigger game, I think the ideas we have here could serve as an excellent baseline.  The full version would include multiple levels in different environments and there would be multiple people to save, Mass extinguisher would have axe as a weapon to attack walls and maybe future enemies, he could use his waterpack as a jetpack to fly upwards, but he couldn’t shoot while flying, and a few different game modes: a horde/survival mode where the objective would be to save the building itself.

In terms of scheduling, we should have done a lot more work up front.  If we figured out how to code the jumping, collisions, etc earlier, we could have included a lot more in the game without much of a problem: a second level, and maybe another person in the house.  We should have had stricter due dates

**Interface/External Tool/Design**

**Aiden Thinn**

*What I did.*

I created all the art assets for the game, such as the player sprite, the enemy sprites, the water sprites, the house shell which included the burning house and the background, and the game screens. I made these assets in PaintTool Sai as well as Adobe Photoshop. Additionally, I also made the external tool, and the manuals for playing the game and using the external tool. The external tool is a Windows Form Application which allows the user to choose a color that they want for the enemy sprites to look like. Once the user has clicked on the save button, a text file named “Color” will be saved to the debug folder of external tool. The text file can then be moved into the debug folder of the main game: FireExtinguisher folder in order for the code implemented in the main game to work. The file path to the debug folder of main game folder is as followed: E:\FireExtinguisher\bin\DesktopGL\AnyCPU\Debug. I also helped with coding for implementing the external tool into the main game.

*What went right?*

The original concept of the game was followed throughout the semester as the game was being made, it turned out to be a 2D platformer about a firefighter who goes through burning buildings in order to save the people inside it and getting them out alive. Though, certain variables such as the axe and different levels were cut from the final product of the game. The art style remained consistent throughout as well and no major changes were made to it.

As for the code architecture, everything went smoothly after the game physics were figured out and the background was ready for floors and fire to be hard-coded onto. Despite the slow progress at the first two milestones, after milestone 3, the team picked up a much faster speed.

There was no conflict amongst the team members, and everyone was respectful when making changes to the game, or when criticisms were given out. The communication also ran smoothly as no messages went ignored and we all gave each other helpful feedbacks.

*What went wrong?*

As stated before, for the first milestones, not much progress was made because the game physics took a much longer time to complete in the beginning. Figuring out the gravity, collision, and jumping were a difficult process, but these were completed once we received help from the professor as well as the TAs.

Particularly for milestone 4, we decided to have animations working for the firefighter. However, this was not accomplished because the Draw() method did not work correctly, and the firefighter texture did not move down with the camera. All of the team members tried to fix this issue but had no luck. Therefore, we had to cut this from the final product since we needed the player character to be fully visible.

Additionally, we lost a team member early on in the semester and our group shrunk down to three people. This made it difficult for us to make our game with every parts of the original concept, and we had to cut back on things such as the axe which would have been used by the firefighter to take down obstacles, and having multiple levels with different areas.

*What did you learn?*

I learned that time management is incredibly important. For coding, we were not able to complete everything that we wanted because for the first milestones, we took much longer than we should have when figuring out the basic game mechanics. As for the art assets, since I am the main and only artist for the game, I had to manage my time well so that all the assets will be available to put into as soon as the code for them were done. Though there was a setback during milestone 2 when I lost all of my art assets due to a corrupted USB drive, I was able to recreate what I lost.

Furthermore, I also learned to communicate and adapt with the team members of my group, and taking initiative when necessary and giving out feedbacks when needed. A good communication between the group mates are extremely necessary, especially for this group, since we all had to take on multiple roles throughout the making of this game. Without proper communication, we may not have been able to accomplish majority of what we originally intended to do.

*What would you do differently?*

I would have managed the time much better, and kept track of the time spent working on this project. We did not do much during the first two milestones because we kept thinking it would not take long to code the basic game physics, which turned out to be a mistake on our parts. The basic mechanics took the longest and we should have finished them up much early on, and we could have been able to include other contents in the game such as a second level.

Moreover, I would also have scheduled more out-of-class meetings since we realized that those meetings helped us out a lot when coding. We were able to give live feedbacks and get help from each other, which allowed us to accomplish a lot more than if we were to work alone separately.

**Architecture**

**Bailey Russo**

*What I did.*

Player’s Process Input, player animations, game and player states, shooting and jumping functions and the Fire and Water classes with Robby, semi-compatibility with Xbox controller.

I initially created the fire extinguisher class to the point where you could move the player around the screen -later updated by the group as a whole. I spent a majority of the time this pass sunday trying to figure out how I could implement the animations, but I couldn’t find a way. So instead I just made it so you could flip the player horizontally. The shooting was one of the hardest parts of this project, it took some time to figure out. I finally found a way to make it so you had to wait until the water you fire disappeared in order to shoot again. Jumping was  attempted by me but not fixed by me. Multithreading doesn’t work in MONOGAME. I believe that I started the fire class with the: damage, tex, and the health. Robby finished the rest of it. The controller input was suppose to be in the game, I started it and worked pretty good, but due to time constraints, I couldn’t finish it.

*What went right?*

There were very few things that went right in this project. First off, the player moving left to right was by far the easiest thing to do in this project. That took roughly five minutes to do. We as a whole had the right idea for every class and how it should be structured. We had get/set methods, draw methods and an organized constructor for each class. The pictures Aiden created were very good, I didn’t expect such well drawn images.

*What went wrong?*

The player animations weren’t working from the start. It was hard enough just to get the player to flip horizontally. The draw method had eight different overload methods, and everytime we would type in the parameters it would give us many red lines under certain parameters. The player would constantly collide with the ground above it and would be stuck there. We couldn’t get the color to work for some time. If you collided with the fire you would turn red and just stay red and wouldn’t turn back to the originally color. Whenever we shot the water, if we clicked the button more than once, the water would reset in mid flight. Lastly, the glitch I talked about earlier. If you hold the left and right mouse buttons, the water stays in the middle and you become invincible. We couldn’t figure out a way to fix this.

*What did you learn?*

I learned how to respect the people who make games more. The progression of making games is a lot harder than I initially thought. You rarely ever go start to finish without changing certain things or going without problems. I was debugging for roughly three hours on sunday to fix an error that someone fixed in three minutes. I believe that the most important thing that I learned is that whatever I imagine in my head, I can make it. That’s the beauty of game programming, you’re suppose to create a reality in which a regular person doesn’t experience in a normal day.

*What would you do differently?*

I would of managed my time more efficiently. At the beginning I guess I thought we had this in the bag. Then as we progressed I started to realize that this is going to be a lot hard than I initially thought. No matter what you do, you cannot delay/put off things that are critical to your game. As a group we were constantly doing 90 percent of each milestone 4 days before it was due. I take responsibility for a lot of that, I shouldn’t of procrastinated as much as I did.