A Brief Chase for a Briefcase Critique

When I began the brainstorming process for creating my game, I knew I wanted to make a side-scrolling stealth game that involved the twist of reversing gravity. I wanted to add things like gravity related puzzles, moving guards, a leaderboard, and multiple levels. However, not all of those things made it into the final version of the game.

The main reason for not everything making it into the game were the limitations that involved manually mapping out the x-range and y-range of the floors and walls of each level. As you can imagine, this took a lot of time. Because my game was (a very small version of) an open-world game, I had to allow the players to move around the map and go anywhere they wanted. This meant bulletproofing every single surface that the player could come across.

Another limitation that the game has is the projectile animation. When the player shoots a dart at a wall, the dart sometime disappears just before it hits the wall. The reason for this is that the darts are moving 16 pixels per frame and so it was difficult making them disappear exactly as they hit the wall. I decided that, after spending a while making the darts more precise, even though they still weren’t perfect, to leave them in the game as a ‘known shippable’. That, in the game making industry, is a phrase used to identify known bugs in the game that don’t affect the gameplay so they are left in.

Most of the major bugs I came across in the game came near the beginning of development and had to do with the movement of the player and the map around him. Because the genre of my game is side-scroller, it made moving around and having items and guards move relative to the map more difficult. For example, when I was first programming in the guards, I had a bug where the guards would move relative to the player rather than the map. So when I would reverse gravity, the guards would rise to the ceiling with me. The way I fixed this was by always factoring in the co-ordinates of the background image when blitting the guards, boundaries, and item.­­­­

A design decision that the players of the game will hardly notice is how intertwined the code relating to the movement of the main character is with the animation. If you look under the hood of the game, you would see that at every step of the way, I slightly edit the animations that the character is doing. For example, at the start of each frame, I set up the possible running and standing sprites. As the character moves, I flip those sprites to face the direction of movement. As the character is standing idle, I flip him to face the most recent direction of movement. And each of these changes happens as the program is processing the movement of the player. The reason it has to happen like this is because at the start of each frame, I have to account for all possibilities

Overall, I think that the areas that my program excels in are creating open-world scenarios and having useful menus. The open-world scenarios portion includes being able to interact with the environment and enemies and the menu portion includes having a menu that gives the user information about the last option that they hovered over and is generally very easy to understand and use. Reflecting back on my game, I think the best way that I could improve it would involve making more maps that the character can go to. Coming up with new gravity-related puzzles that I could incorporate into new maps accompanied by the amount of time it took to create them was too hard and so I wasn’t able to make more than one large map by the deadline. That being said, I am very proud of the game I have created, and with more time, I know I could improve on it.