**Name:** Tanat B.  
**Group Name:** Palette Pals  
**Role:** Gameplay development

**Features worked on:**

Because I had more experience with the infrastructure of MonoGame and XNA, I started work on creating basic structures of the GameObject class and part of its children classes. I worked on the game loop which does the following: updates the positions of the characters, check for collision, fixes the position due to collision, takes input, and then updates velocity and acceleration based upon input and collision. In addition I wrote helper functions that checks for collision and fixes the position of the player. I integrated the camera movements and the input handling to coherently work with the gameplay loop. I worked on the StageManager which manages the main game and all the objects that are needed. I also helped document the game through comments and summaries.

**Bugs and Issues:**

There are a notable collection of bugs and glitches that is involved with the gameplay loop. Firstly, the game isn’t set up in a grid system. Although movement and input works perfectly fine, the player’s visual representation is not too accurate. When the player is positioned on top of a block, he/she will constantly gyrate atop the block as he/she constantly falls through and gets his/her position corrected. To combat this issue I tried to project the character’s position to check for collision before updating its actual position. This seems to solve some of my issues, but errors still remain within the system. As a result of this constant downwards movement, the game registers the character as if he/she was falling, disabling her ability to jump. Otherwise, jumping would work perfectly fine. When the player collides from a bottom of a tile, he/she intersects the tile for an instance of time before slowly sliding down towards freedom. The cause of this issue seems to be due to the collision detection function and position fixation function.

**Changes:**

So far, the timeline has not changed. We have accomplished all the points that were listed in the according weeks, but the level of accomplishment has yet to reach the highest level of quality. Disruptions due to GDC were minimal, as minimum requirements were reached. However, rather than being satisfied with achieving minimal standards, we seek to hasten up our schedule to accomplish notable progress. We plan to achieve stellar results by investing more time and effort into each week.

Notable changes to the game as a whole exists. This does not drastically change up the timeline because this pivot does not pose a different amount of workload. Firstly, we will change from using the RGB color scheme to the RBY color scheme. This is because it makes more sense for the character to use the color logic seen in mixing paint. Secondly, the puzzle dynamics will change. The player will no longer switch his/her color at a palette table, but he/she will have to drain color from an enemy (defeating it in the process). This change in puzzle elements does not change the due date of each item on the timeline. It just merely replaces the palette table that was initially planned.

Sam Heckle

Pallet Pals

3/9/2016

My role for this milestone was the tool builder. I created the class that implements the external tool that reads our map. Another team member, Alex, built the feature that allows for us to input what we want our map to look like, and it exports that as a text file. From this, I take the text file and read it from its current location with a BinaryReader. The first two items in the text file are a string and an integer variable, and the rest are simple characters that determine what each floor tile looks like. The BinaryReader reads the first two items directly, and then it goes through the entire file character by character until there are none left. In order to do this, the reader checks the current position and compares it to the length of the file. Originally, I ran into an issue to actually find the text file, but this was resolved when I found the full file path into the folder. Then, I had additional problems actually checking if the document was finished, but again, I ultimately ended up resolving this quite quickly when I realized that BaseStream has properties for both position and the length.

As of now, we do not have bugs that appear to be major issues. Other than working out minor kinks in the actual code by testing each class individually, there has not been a bug that has not been fixed quickly.

There have been two minor pivots in our design implementation. First, we decided to change from RGB color to RYB color. By doing this, it becomes an additive color game, which makes it more simple in the long run to change the colors in the actual game because the coding will be much simpler to implement. Secondly, we believe it to be a more interesting game if we changed our table idea, where the character can change colors at a standard point in the game, to a gathering idea, where the character gains ammo from enemies that are killed. This way, the character is limited by the amount of ammo they can use, and also colors cannot be generated by the player.

Lastly, there have been no changes made to the UI, thus keeping it the same as milestone one.

**Name:** Alex Hazen

**Group Name:** Palette Pals

**Role for this milestone:** Designer, Map Maker/Tool Builder, basic asset production, and clean up some of the Character and Object Implementation.

**Features I worked on:**

One big feature I worked on was the external Stage Builder/Creator External Tool for anyone who is designing or creating a stage for the game. This “stage builder” is a Windows Form Application that allows you to create a text file that maps the components out for the game map. If you were working on a stage at a previous time you can also load in your previous work into the forum application and edit it using the “Load in File” button. The application also has a nice table of contents to the left allowing editors to know what each character loads into the game as.

There were a couple issues that I had when working on this forum application. The first was figuring out how I could create a text box that would allow the designer to work left to right instead of up and down when typing out the stage. Originally I had the application take only one part of the stage at a time, however, this was quickly thought as inefficient for the developer and the StageReader class for the game so it was quickly changed. Also another problem was allowing players to override a previous text file or someone’s progress on a stage if the file already existed. To combat this issue I made it so you first have to type the file name before the “load button” is activated and made it so you had to click “load file” before you could access the button that saved/compiled the text or edits for the map into a text file.

One other thing that I worked on was finishing some of the first art assets for the games. I edited the player character and enemy characters Tanat made and I created 5 different ground tiles for the game.

**Bugs/Issues:**

There are a number of small issues that have yet to be fixed. The first is that when you load in the text file with the external tool it also loads the first two lines of the text file which are meant to display the name of the stage and level number. I have yet to find a solution for this. Also when loading in text to the text box it also creates a blank line at the bottom of the text box. This means if you load in a file you have to delete the first two lines (stage name and level number) and delete the extra blank line at the bottom otherwise it will duplicate the stage name and level and save an extra blank line. Currently there has not been any bugs that I can tell that were introduced when I integrated my code to the overall game project.

**Changes since Milestone 1:**

There have been a couple changes about the game, mostly design changes. The first was the change that we made concerning the color wheel that is used in the game. We decided to go to switch from the additive color wheel to the RYB (red/yellow/blue) color wheel. This was made due to the character being an artist and using paint, not light, to mix colors. Another thing that has changed was how the character will be switching colors. Instead of having a table where the main character can switch his paint he will be getting the paint from defeating enemies (by swiping them with his paintbrush). This means we no longer need a class for the palette table and instead we will need a new method for gaining paint (in the player class).

Seth Weidman  
IGME 106-02  
Professor Bierre  
March 20, 2016

The role I played in this milestone was to properly handle the input for the game (player movement, projectile firing, etc.), as well as to create a “camera” of sorts that follows the player as they move backwards, forwards, and when jumping and falling. Also, like the rest of the team, I assisted in adding comments to the classes and fixed errors where I could see them.

The features I worked on the most for this milestone were within the classes of Character, Player, Enemy, GameOBject, Game1 and Camera. Within them I added proper code to help establish a connection between user input and player movement, as well as enemy movement. Using a WASD control scheme (minus the S - No ducking), taking in the values the player inputted and implementing them with the actual movement variables was difficult, mostly in the fact that we had a few Source Tree problems. To explain, let’s just say I made some edits(implementing the camera movement code from my test project to the game project) and didn’t push them until after someone else made edits and pushed them, so we some loss of data. It wasn’t completely disparaging, but I do feel ashamed on how we had to go back and redo some work due to my laziness. Going back to thing I did work on, the Camera; it was fairly easy creating it in my own project, but putting it into the game project and having it work with all other classes in unison was not as easy. Still, in the end, I think we are in a good spot. At least now I know what not to do, and know what effort and care I must elevate myself to.

As mentioned earlier, we did have a pushing and pulling issue that inconvenienced us for a day, but fortunately I believe that issue was fixed. And the integration of one person’s code onto another’s is bound to have at least some issues. I can’t think of anything off the top of my head that was truly devastating in that respect (besides the above example), because we have all been just fixing errors here and there. If we all work together, and play off each other’s strengths and niches, we can ‘roll with the punches’ so to speak, and quell these bugs without much backtracking.

Yes, there have been changes since our last milestone. In terms of input, I originally thought we would be using arrow keys to move, yet this is changed now, due to consideration towards the ease of a player’s hand to rest on that side of the keyboard. We also have now cut one of our classes, the Palette Table, to condense our work load. Originally, we would have our player face a puzzle that would require the use of a Palette Table to change the colors of our player’s projectile. Now, we have the player able to change the color by shooting at an enemy of the color of desire, which I think is a great idea in reduction of work to still perform the same task. We have also added a Camera class to our project to simplify the movement of the ‘camera’ following the player. Other than that, the class structure/architecture has remained relatively the same, as well as the look of the game, (save for a change in the player’s colors - i.e, a different color swatch just because we thought it more vibrant and well balanced than the previous swatch), and the timeline, to which we are on track.