Alex Pearce – Team 3 Musketeers

Game Development and Algorithmic Problem Solving II

Professor Kevin Bierre

March 19, 2016

1. Name: Alex Pierce Team: Team Three Musketeers
2. I worked on character and object implementation.
3. I set up all of the current classes and their inheritance. I also made the screen wrap and fixed the bullet firing.
4. Originally the bullets were being created in the corner of the screen and not moving after the space key was released. Once I fixed that I didn’t notice any other bugs.
5. The architecture has not changed since milestone 1 and we have stuck to the original plan.

Niko Bazos – Team 3 Musketeers

Game Development and Algorithmic Problem Solving II

Professor Kevin Bierre

March 19, 2016

1. Niko Bazos, Team 3 Musketeers
2. My role for this assignment was Game View.
3. I worked on the Game1, GameObject, and Bullet classes. In the Game1 class I worked on: providing a menu screen that displays what certain key presses bring the user to, programming the movement and displaying the character in the Game state, and starting a projectile algorithm (that was later fixed by teammates) as well as an animation finite state machine with corresponding draw methods that are still in progress. In the GameObject class I worked on: adding a health attribute and property to the GameObject class due to the fact that we decided all objects in the game would have health and could be destroyed. In the Bullet class I worked on: creating a Collision method that determines if a bullet intersected with an object, and if so, subtracts a certain amount of health using the Health property implemented in the GameObject class. I also created the character and bullet assets for the game as well as went through the code to make sure all coding standards were met.
4. When I was working on player movement using rectangles I encountered a problem where I had all the correct movement code but was referring to a null rectangle as I had not created a rectangle for the character. I had some other minor issues such as not selecting Content or Copy Always, but other than that nothing major. To track down and fix bugs I found using the debugger and commenting out certain pieces of code to be very useful in order to pinpoint where and what I had to change.
5. The design, architecture, look, or timeline has not changed since milestone 1 with the exception of the movement being changed from WASD to the arrow keys.

Dezmon Gilbert – Team 3 Musketeers

Game Development and Algorithmic Problem Solving II

Professor Kevin Bierre

March 19, 2016

For this milestone my role was primarily **Gameplay.** I worked primarily in the Game1 class and worked on the game states and decided the factors that would change those. I used a finite state machine and use a switch statement that used the enumeration variable as the value to check. Upon initialization, the game state is by default set so that it will always go to the Main Menu. From the Main Menu, the player can press certain keys in order to change the state. For example, pressing the “S” key starts the game by switching the state to the Game state or pressing “C” changes the game state to Credits so that the credits menu is drawn.

I also created parametrized constructors and attributes for certain classes. I had to code the parameterized constructors so that the game would compile so that I was able to test the change in states and make sure that they worked properly.

Currently, there are no bugs in the part that I worked on. The only issues that I want to work on for the next milestones are related to making the game playable.

Since the last milestone, the only thing that has changed as far as my role has been the controls. Instead of WASD we are going to use the arrow keys for character movement.

Alex Martinelli – Team 3 Musketeers

Game Development and Algorithmic Problem Solving II

Professor Kevin Bierre

March 19, 2016

Role: External Tool

Our external tool is a map builder that uses a Monogame window to preview the map and a Windows form to create objects. The objects that are shown on the Monogame window are objects of the “Scenery” class. These objects have a rectangle object, a loot chance, a health value and a texture. These values, with the individual X, Y, Width and Height components of the rectangle, are set using the form.

The form uses numeric up/down boxes for the int values. While the boxes do not initially use ints, the numbers are rounded before they are stored for the creation of the object. For the texture chooser, I used a combo box with the names of the textures as selectable strings. When a new texture is chosen in the combo box, a preview of the texture appears below. To create the object, the user must hit the confirm button. The button then checks for a few things. First it checks for the rectangle values. If the value of any of the boxes equals zero, then the text of the button will change, stating that they must be filled. Then it checks for a chosen texture. If no valid texture is chosen, then the button’s text changes, telling the user that they must choose a texture. Finally, it creates the object and checks its rectangle for collisions on each of the previously submitted objects. If any conflict, the button says so. If the object is created correctly, it is then added to a list of objects. This list is then saved to a file using the “Save Map” button. This feature is still a work in progress, however, and will be worked on further in the future.

Each scenery object, in addition to having a rectangle and texture, have a health value and a loot chance value. These values are more important to the gameplay than the actual map tool itself. Each object is given a health value, meaning that when hit with a bullet, it will lose health. When its health reaches zero, the object is drawn as rubble instead of its normal texture and, based on the objects loot chance, it drops a piece of loot (i.e ammo).

There were quite a few challenges when writing the code for this map tool. The first and most problematic was drawing the new objects as they were created. The problem was that I was setting the objects textures in the initialize method of the game class. This was bad because the texture2d objects I was using weren’t being initialized before the objects and also because new objects wouldn’t be assigned a texture. As a solution, I assigned the textures to each object in the update method. Another problem I encountered was with the texture preview. I just couldn’t get it to show up, but after a bit of research and putting the images in every directory in the game files I got it to look the way it should.