**Work in Progress Report 3**

Major developments/breakthroughs(reference specific code please):

-creating enemy spawners that can be replicated easily

-separating common code between enemies and the player and placing into an entity class which is then inherited by the enemy and player classes

Major Challenges/setbacks( reference specific code please):

-

Any modifications to your specifications/release schedule:

-no modifications

**Description of your scratch/test program:**

**1. (Rueban) 893SpawnerScratch**

Describe the generic concept you needed to test out:

-making enemy spawners that can be replicated and relocated to any part of the map that only spawns enemies within a certain range

-separating common code between enemies and the player and placing into an entity class which is then inherited by the enemy and player classes

Source any web site/book that helped you with that concept:

Main Source: Don’s code for the spawner main body as well as grabbing map data from tiled

Describe the code and the lesson that you learned from it:

* To grab the spawner locations from tiled without having to determine which objects are spawners and which aren’t, I created a new object layer specifically for spawners. Then all the objects in the enemy spawner object layer were cycled through and each of its’ coordinates were put into an array of Vectors. The array is passed into the scrTest class then each set of coordinates creates a spawner. The spawner then gets the distance between the spawner and the player, and if the distance is below a specified amount, it will spawn an enemy. The number of enemies spawned can be varied based on the length of the array of enemies. Each enemy is then updated as part of the spawner class.
* A new class called the entity class was created which contains all the common code between the enemy and player classes. The enemy and player classes then extended the entity class. I learned how inheritance can be used to reduce the amount of repeated code significantly.

Describe any challenges that you enjoyed in integrating this scratch code into your major project:

-the collision detection was changed while I made the scratch so I had to tweak my code a little to have it work in the final project

**2. (Don) Firing**

Describe the generic concept you needed to test out:

I wanted to make the player fire a bullet at the position of the player in the direction that the player is facing. The bullet would have these properties:

* Have a high velocity
* isBullet = true (would allow for continuous collision checking, higher accuracy than with it false)
* Have a cooldown period to disable constant shooting
* Not collide with any part of the players, but collide with everything else
* Adds to an arraylist
* Easy to expand

Source any web site/book that helped you with that concept:

Group Indexing: <http://www.gamefromscratch.com/post/2014/09/25/LibGDX-LibGDX-Tutorial-13-Physics-with-Box2D-Part-4-Controlling-collisions-using-filters.aspx>

(more specifically the very bottom, where it talks about the properties being changed based on whether the value is negative or positive)

Bullet code:

I wrote shooting code a year ago. I replicated it. I’m the source.

Describe the code and the lesson that you learned from it:

A new class is created named “Bullet”, which contains all properties of the bullet including shape, velocity, and creation. The class follows this order:

1. Add a new instance of itself to the arraylist
2. Create the body
3. Check the direction
4. Apply velocity based on direction

The rest of the work is handled by the world.

In the contactListener:

1. On a contact, check whether one of the contact fixture is a bullet
2. If bullet, check what index
3. Save index to an integer, set userdata of bullet at index to a flag

Outside the contactListener:

1. Loop through the bodies alive in the world
2. Check the bodies for the flag in the userdata
3. If the flag is true, delete the body and remove it from the arraylist
4. Reset integer

Describe any challenges that you enjoyed in integrating this scratch code into your major project:

It’s an entirely independent concept from everything else. There were no problems adding it to the project.

**3. (Kevin) BulletVsEnemy.**

Describe the generic concept you needed to test out:

* Keeping track of collisions between two ‘entities’

Source any web site/book that helped you with that concept:

Describe the code and the lesson that you learned from it:

* Setting a boolean for when the bullet should be deleted (ie. it collided with an enemy) and deleting it in the ‘clean’ routine Don set up.

Describe any challenges that you enjoyed in integrating this scratch code into your major project:

* It got messy because of basically pasting everything from the other member’s repositories, this caused a few bugs but they were easy to fix

With each WIP, you will be submitting EVERYTHING. Organization is key. When I go to the groupwork folder**, I should see your project submitted in the following format:**

YourLastName: Under this folder will be the following folders:

**Asana Specs**: Your Asana calendar will have a task that contains a github link to your project and scratches. Please add any comments within this task that can give me a better understanding, like : “It does not work.”

Even if you provided the link to the same project in a previous task from a previous month – go big – add it again.

**Documents**: It will hold all of your documents: journal, WIP, Specs, Release schedule, list of sources, and all the other documents that will be submitted in your final project.

**Releases**: There will be a folder for each release, with one folder CLEARLY telling me that it is the latest, stable release.

**Scratch**: There will be a folder for each scratch concept that you tested before you integrated it into your final project.

**Peer Assessment:**

Don 100

Kevin 100

Rueban 100