Worklog summary week 4

Angelo Anthony Rettob  
181494, IGAD Programmer

[**Monday**](#_fviv1098jcwh) **3**

[**Tuesday**](#_trj5e04h9c04) **4**

[**Wednesday**](#_3emke4a0r2nf) **5**

[Schedule](#_5yji2agmae67) 5

[Events](#_1710ao6fob1q) 6

[Collision response](#_bpeau7re1m5z) 6

[Workshops](#_5926fadsvxmx) 8

[**Thursday**](#_jckesbc422t0) **9**

[Schedule](#_actxtz1yo7c1) 9

[Implementing entity classes](#_prbnaqyr2c3r) 9

[Implementing object collision response](#_4lbfhdb57fkl) 9

[Implementing entity spawn classes](#_b9uji54o09mn) 9

[Implementing AI pathfinding](#_lbus1jcekl0x) 9

[Events](#_my8zl476crzl) 9

[**Friday**](#_87qdbv6tz3k3) **11**

[Schedule](#_j1e3tx5cb7kp) 11

[Events](#_8skreuoao3aq) 11

[**Saturday**](#_fnvts8atimyh) **13**

[Schedule](#_lurfnihio8b6) 13

[Events](#_3m2nr9bhrhfr) 14

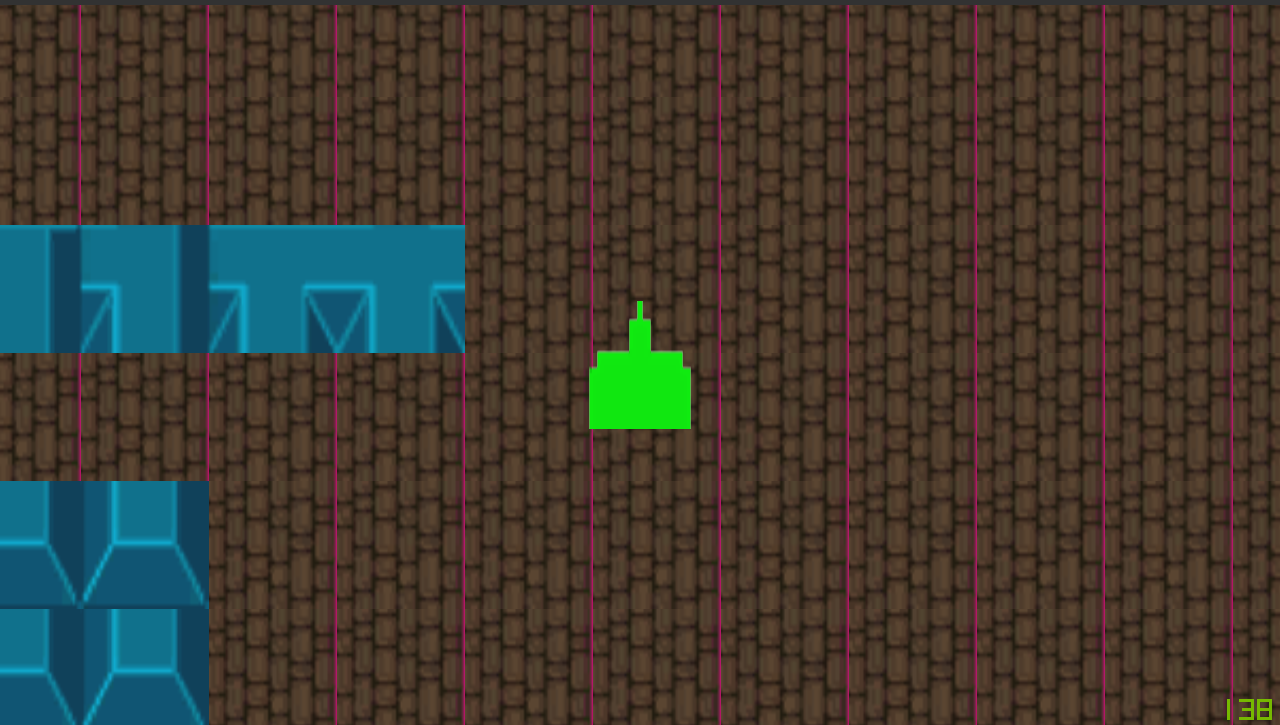
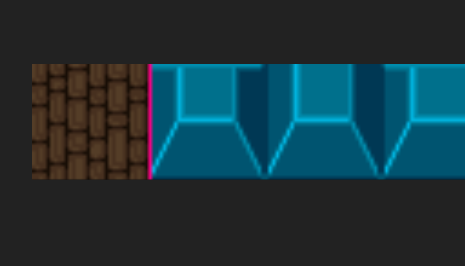
[**Sunday**](#_wp1wiwkd9g2o) **15**

[Schedule](#_6ib8d138irqz) 15

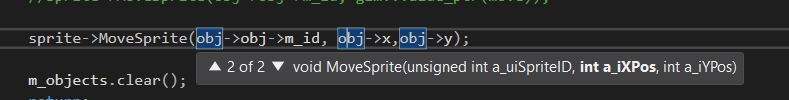
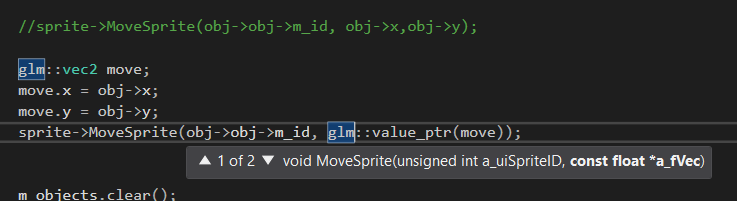
[Events](#_lynm5fqc91fz) 15

# 

# Monday

* Converted map scrolling to camera scrolling.
* Learned about matrices and that GLM has the functionality to modify matrices.
* Helped peers with XML parser. It helped me be able to explain my code more clearly.
* 
* 
* Phil told me about float rounding issues. I should look into point filtering/sampling.
* Couldn’t find anything after searching for 1 hour. Will look into this another time.
* Implemented the physics engine.
* The framework doesn’t blend setColor function but overwrites the whole color.
* Brian told me that I don’t have to worry about the rendering issues I’m facing.
* I have written a physics engine that as of now can only detect on what tile we are at.

# Tuesday

* I’m going to implement collision response. The way I’m going to do it is by using AABB collision because this game is based on a grid. I normally use circular collision but by using AABB collision, we can use the same collision for every object.
* I’m thinking of using a constant size for every object, 32 x 32. The only problem with this is that keys are smaller than 32 x 32. I will see how badly this affects gameplay.
* I noticed a bug with the physics engine. For some reason, it accelerates more when I’m subtracting positions than when I’m adding positions. [See this](https://drive.google.com/open?id=1tTR3Cj2KhtarEcoCTBOQc1wTMhPY4UR6).
* I have found the solution.  
  This makes the object move at a different pace. When using this type of the function, the positions get converted to integers. Because integers round down, we move faster in negative directions. To use the second version of the function, we have to use the other version that is provided.  
  This makes sure that the data stays a float. Value\_ptr just maps the GLM vector to a pointer. Just to stay in check with the rest, I also used this style (although I could easily just make an array and pass that in, I think this is the intended way). We learned something new!

# 

# Wednesday

## Schedule

I’m going to attend 3 workshops.

* Getting into game careers  
  This sounds like an important workshop because the information that I learn from this is very valuable for when I’m going to work in the industry. I’m already working on a portfolio, this lecture should only improve it.
* Introduction to DirectX 12  
  DirectX is something that I always wanted to know about. Reading documentation and tutorials sure is nice, but by getting information from a trustable source is way more pleasing. DirectX is something that is used very much and I think it’s certainly something that I can use in my own projects, or at least learn from.
* Expert group code review  
  I have been complaining about how I couldn’t plan code appropriately. I always had questions about my code and how I format it. This is the perfect workshop to address that issue.

I’m going to work on the collision resolve logic. Right now, collision is detected just fine but the user can’t walk against a wall. It just stops the user from moving, this is solvable though, I just have to calculate the new position instead of sticking it to its own place.

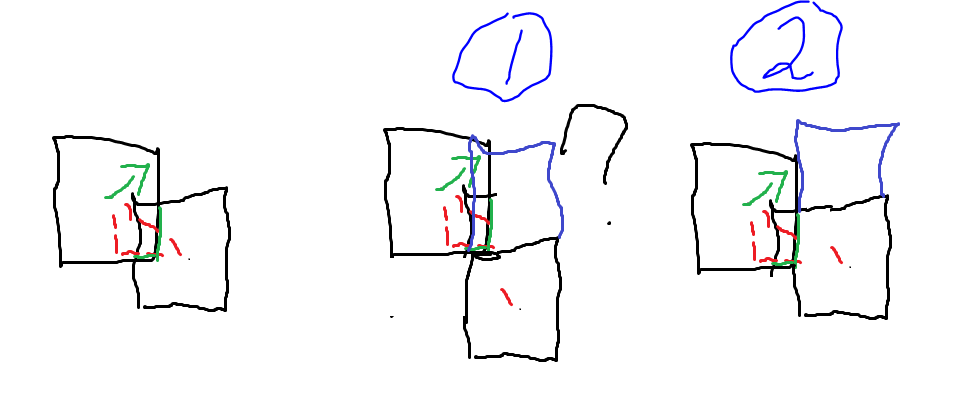
According to my planning of this sprint, I should also be finishing another part of the PDP. I’ll be doing that today.

## Events

### **Collision response**

(Excuse my paint skills, this was made with a touchpad)

How are we gonna respond to this collision? How should the physics engine know if we need case 1 or case 2? The way I’m gonna do it is by doing 2 checks in both directions. Look at this.



If there is an additional block and we move the object in one axis. It can check if there still is a collision. If there is, it should be the wrong direction. In this case, the response should be case 2. Let’s get some pseudo code going:

Calculate delta between origins (which should be center)

Calculate position to touch the edge of the tile

Check if it collides when moved, else use another axis

This looked like a mess when implemented though. Let’s take a different approach. By calculating the distances between the objects, we know on which side the objects collide. We can just snap the object back in the shortest distance that it has collided

Check which delta is smallest

Move in that direction

This doesn’t work, it warps my player for some reason. Let’s just implement a simple method that has to work for sure. If it’s slow. I’ll just optimize it later.

Get delta positions

Check if axis collide by only checking one axis at a time

If axis collides

Snap to the edge of the tile

I learned something very weird. My character keeps teleporting and I didn’t know why. It only happened in the positive directions. Have a look at this

See that value at the end? That’s **way** too big. So why does this happens? Look at the conditional operator.

There is our problem. The cellHeight variable is an unsigned integer. It can’t be negative. Because of that, the value wraps. Can’t believe a data type mismatch made this part of the game so hard for me, but now I know what to look at first when a problem like this reoccurs.

It is almost finished, but something is wrong, see this

Those edges are unacceptable. After a bit of debugging it appears that because of my early out condition (when one collision is detected, no subsequent collisions have to be checked) resulted in this problem. So I just made the program check all collisions, this works but we lose the benefit of leaving the collision detection earlier. Then again, this is probably what they call prematurely optimization. Anyway, I shot myself in the foot and should try to avoid doing that next time.

### Workshops

The game career workshop was very informative. It told me what the companies are looking for. The top 3 things for programmers are:

* C++
* Maths
* Portfolio

They also told us about their program called searching for a star. This is a competition in cooperation with companies. It can be seen as a scouting program. Very interesting indeed.

The DirectX 12 workshop wasn’t what I thought it would be. Sure it was informative, but it was very specific. At least, that’s what I thought for an introduction. It is my first workshop that wasn’t recommended. I noticed that because it was way less crowded.

In the end, I decided to go home and work on my projects. I skipped the code review expert group because I got a bit tired and decided to go home early and work on my project. I also don’t have a lot of code that I have questions about. Next week I do and I can ask a bit more about my code.

# Thursday

## Schedule

### Implementing entity classes

I’m going to get a lot of the objects in the game going. We need to implement plenty of them. Before I’m gonna do that, I’ll need to get the right textures in my project. I will prep textures for this and write a simple animation manager for walking animation purposes. I won’t be needing a state engine because enemies only walk and attack whenever there is a player in the line of sight.

### Implementing object collision response

I’ll try to implement at least every entity in the game. Not yet their behavior, only their textures, and boilerplate code. I will be implementing their collision response which should be the same for every entity. Just damage each other. Although entities should only damage players. Because of this distinction, we will only differentiate in object type by player or enemy/monster.

### Implementing entity spawn classes

After that, I’ll write a class for the spawners. The spawners will be objects. I actually found a great read about this [here](http://gameprogrammingpatterns.com/prototype.html). I’ll be implementing this for my game also because it gives me the opportunity to define al all my enemy spawn methods in one class. I’m gonna turn it into a kind of factory pattern.

### Implementing AI pathfinding

I’m not sure about this one. But I did do an extensive research into the pathfinding this game offers. And it’s very simple. You can describe an enemies (except for the Lobber or something) behavior this way:

If no object next to me and line of sight to the player is parallel with world axises

Shoot

Else

Calculate the fastest axis towards the player

Walk that axis

Yes, that’s it. Notice how the enemy only check if there is an object next to him. This can cause some funny behavior where the enemy shoots another enemy. I call that a gameplay function.

## Events

After 1 hour I’m still modifying sprite sheets. The program I am using is just way to slow. I’m gonna use paint dot net. UPDATE: I like paint.net

Sprite sheets are ordered in 3 files. One for animations, one for static and one for the tiles. There is no apparent reason as to why I split these up, but having animations separate from the static allows me to define themes and animations counts without having to worry about the other sprites. It just saves me on calculations.

I have added a function to the map to get the surrounding tiles. This is because the Spawners need access to the map to spawn their monsters.

Let’s get the objects into the map. Because we have moved the logic of movement inside a physics manager, we can easily add logic to move them in the map also. We just calculate the position in the map after the movement and add their address to it.

There is one thing that I overlooked. To get the logic of objects inside of the map, I first have to get the objects inside the map. Do I first have to export tiled into, well… objects. So my whole schedule got a bit messed up.

# 

# Friday

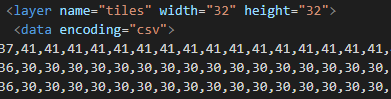
## Schedule

Today I’m going to implement everything I made up yesterday. I would say that Thursday was much about preplanning the project. I had planned way too much for that day so we’re gonna finish that today.

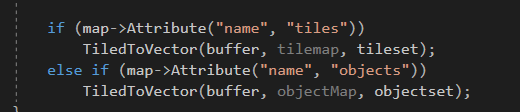
The main problem I faced was that I wanted to move the object spawn functionality inside the map class instead of the game class. This required me to move a lot of functions. For example. The camera class needs the player’s position but the player is spawned by the map so they need to communicate with each other. The map can only create objects with help of Tiled, so this means I first have to implement Tiled object generation. For that, I need to have a sprite sheet that represents the sprites in-game.

## Events

We first need to be able to load objects. To implement this with tiled, we need to call our buffer converter again. Very easy. There is one problem. How do we know with what tileset we’re dealing?



We can make layers in Tiled and give them a name. With tinyxml2, we can easily search the element if it contains a certain attribute.



Just like that. There is only one problem. As you can see in the previous picture, the one above the one above, the values are quite high. These values are supposed to represent tiles and we use the id to calculate our UV coordinates, so why are they so high?

It’s because Tiled won’t decide where your tiles go into. We have two different tilesets. One for tiles and one for objects. This is so we can convert the two objects into their own. What I mean by this is that tiles stay in tiles type and objects stay in objects type. There is not much reason to do this other than to not have object types called tileUp, tileDown, but instead keep them in their own enumerator.

So how do we convert these ids to their correct ones?



Tiled calls this firstgid. Not sure what it means but my guess would be the first global identifier, either way not important we just have to use it. This value can be seen as an offset which Tiled uses to make sure that every tile has a unique id. We can for example use objects in the tile layer, but this would crash the Tiled converter I wrote (unsigned integers would wrap) so don’t do it. Instead, we use the offset to offset our value back.

That should be all for the TiledConverter class. It should actually be finished for good now.

Let’s work on the map now. We need a way to convert the objects ids into actual objects. Let’s make another Library.

One big problem that I faced during the creation of the object Library is that I had to convert every ID in Gauntlet to an ID in the object type. This means that our object type gets bloated with enemy breeds. It’s quite ugly, but it does the job. In a large project, I’d probably won’t use it (but I would probably use string hashing for object types instead of enumerators).

# 

# Saturday

## Schedule

I didn’t have the time to finish up everything yesterday (only had about 3 hours left when I came home) so I postponed everything to today. One thing though, today we celebrate my moms birthday, so I’m not sure how I’ll be doing with time today.

Our objective is to load spawners and spawn objects in the game. But to spawn any object at first, we need to put it in Tiled. So this is the sequence of operations.

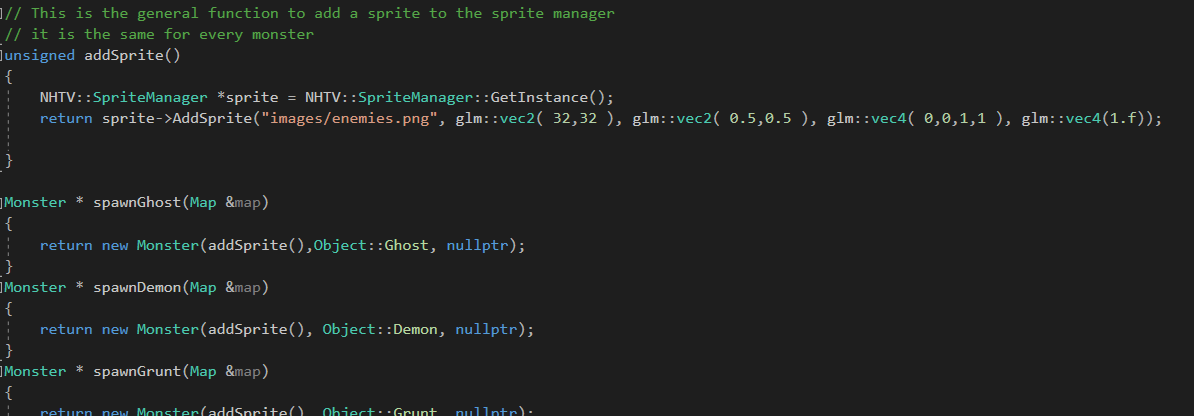
* ~~Make resources for Tiled~~
* ~~Load resources made by Tiled~~
* Modify and use the resources made by Tiled

So how are we gonna use the resources made by Tiled?

Every tile has its own id. We have to map this id to an object. To do that, we write a kind of map. Something that takes an id and returns an object.

How are we gonna implement spawners? I’ve written about this before [here](#_b9uji54o09mn). I’m gonna talk a bit about the implementation here because I now know a bit more about how the code has structured itself.

It is quite simple. We will map all the spawners also into the object library. I have defined functions that define how monsters are spawned.



This keeps all the spawn logic in one file so that I can easily modify it if needed. A spawner will hold this function it will call it when an object has to be spawned. The nice thing about this is that these functions can be used by the object library and by the spawners.

## Events

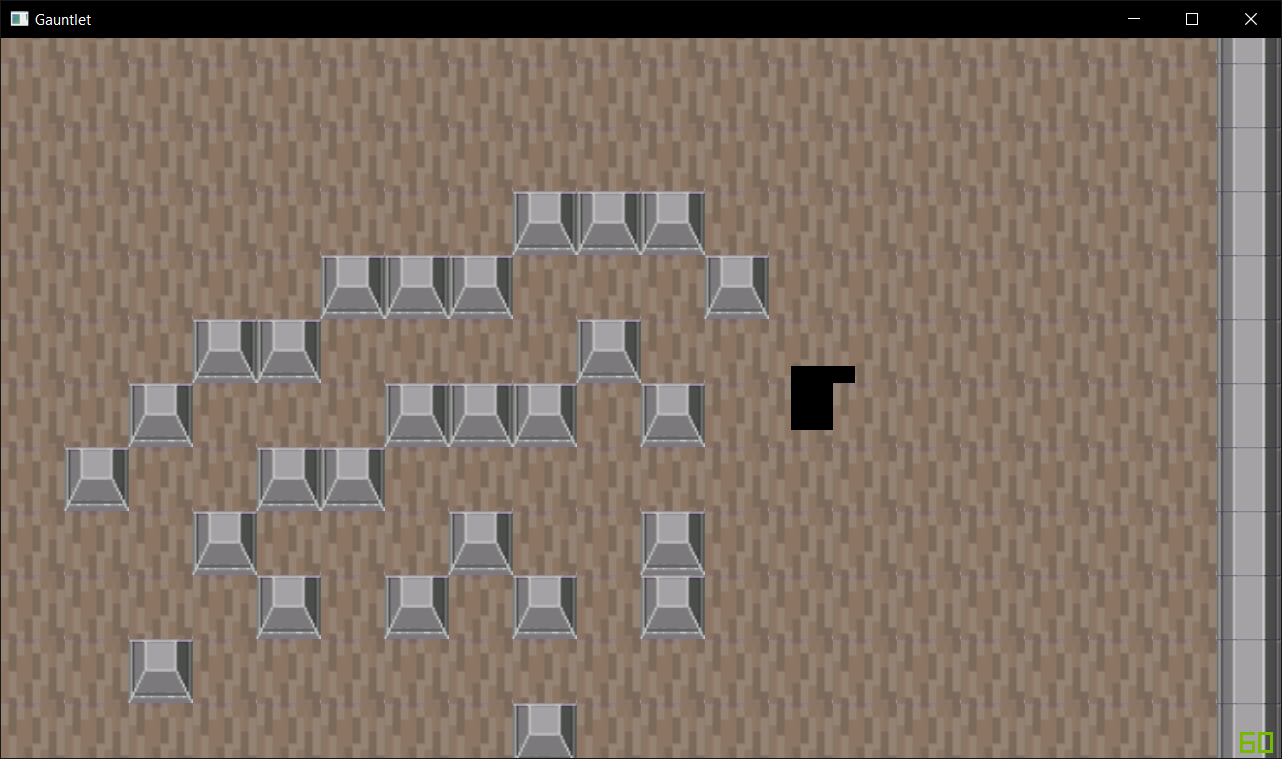
### Mapping ids to objects

Mapping the object was nothing special, it is just an extension of the tile library. I did notice that I could move the logic from all these libraries into one library. It would give everything a more unified design which is nice because they both do the same thing, just on different types of objects. I decided not to do this because I like the distinction, I also have that distinction in my map.

On a late note, the object library needs some internal variables that the tile library doesn’t need, so the distinction isn’t too bad.

*I mean just look at the size of that thing, isn’t there a better way to do it. (Well, of course, there should be, I just haven’t figured it out yet)*

Now that we got our switch, we will fill it up with nice sprite creation calls

Half of my object kept disappearing. 

This is because I drew my tile and then the object on that tile. Because we calculate the object tile position to the nearest tile, it got overwritten. A simple loop that calls the draw code twice fixed this.

I’ve finished up the mapping from Tiled to objects.

# 

That’s pretty nice.

I finally implemented something without breaking all of my code! It’s the spawner class

I also added an animation class.



# Sunday

## Schedule

Today will be a lot of finishing up the work log and writing another chapter in the PDP. I postponed that the previous week so I really have to catch up with that. There won’t be any programming today because I really want to ask about feedback before I will write more evidence.

## Events