Final Project Features

1. libGDX + Box2D

libGDX was used to render graphics and handle user inputs. Box2D was also included in the project to manage collision events between objects in the world.

2. Multithreading

NPCs in the world were multithreaded (check `NPCUpdateThread.java` and `update` method of `GameScreen.java` file). I attempted to multithread several other functionalities, such as parsing multiple layers of data from a `.tmx` file, but libGDX does not allow multithreading graphics and audio, i.e. they all must run on a single thread.

3. JDBC

A database was added to store (in respective order) usernames, number of NPCs destroyed, playtime in seconds, and date and time last logged in.

How to Run Project

Import project into Eclipse and run the `main` method in `Metroid-desktop/src/com.ak4311.metroid/DesktopLauncher.java`. You will be prompted to enter a username. The application will open after entering a username.

Gameplay Functionalities

Controls

A/D – Move player left and right respectively.
W – Jump/Unmorph.
S – Go into morphball state.
J – Fire cannon.
K – Look up.

When the game loads, there is an upgrade on the left side of the map. Once it's picked up, you can hit S on your keyboard and go into morphball state. This allows you to traverse through 16x16 spaces. Hit W to unmorph. When you destroy an enemy, your score will increment by 1 and be saved to the database upon closing the game. Upon receiving damage, the health counter on the top left will update. Currently, there is no game over screen; the console will just print "Game Over!" and you can keep playing.

I think the functionalities were the most challenging part for me to implement. There was a lot to learn on how libGDX and Box2D are implemented. For example, I initially just had a hitbox on the player, but found the player would get stuck on walls. To solve this, I added sensors so when one of the sensors touches a wall, the player's friction is set to 0 temporarily and they'll just slide down. Two more notables challenges were: implementing the behavior for walking around walls for the Zoomer NPC, and solving the issue where entities would get stuck on the edges where two tiles connected (solved using ghost vertices).