Eric Gonzalez
CSC 413

Lazarus Game Documentation

I will be using the wingman game code as the skeleton for this game. I obviously have to remove some unnecessary objects and classes. Some of the classes I will be removing are the weapon classes since there is no weapons in this game, also with that comes removing the bullet classes as well. I will modify the classes named BossEnemy, SideEnemy, BackEnemy, and Shootingenemy, to a corresponding box because in this game you can assume the enemies are the boxes trying to squish you, so maybe I change BossEnemy to MetalBox and SideEnemy to CardBox and so on. I would obviously need to change some of the in-game mechanics but I would also have to keep some, using reusability. Here is a breakdown of what the major classes will do in my program:

LazarusWorld: This class extends GameWorld which implements Runnable. It is the class where we put main(), do all the image importing, objects declaration and timeline control, and maintain some static variables that the other class can use.

GameObjects: This class has all the data fields and behaviors that are shared between the game's objects. This will include the object image, position, speed, and size. This class will also have update, draw, setter, and getter methods.

Box: The Box class will be abstract and extend GameObjects class and implement Observer. This class will have methods and data fields to keep track of the strength/density of any given box. The idea is that the box with the highest strength will crush the weaker ones.

CardBox: This class will extend Box class and have the lowest strength.

WoodBox: This class will extend the Box class and have the third lowest strength.

MetalBox: This class will extend the Box class and have the second lowest strength.

StoneBox: This class will extend the Box class and have the most strength.

StopButton: This is a class I must add. This class will inherit from GameObject and will be used to stop the boxes from falling and consequently win the current level.

Lazarus: The Lazarus class will extend the GameObjects class and implement Observer. It may have additional an additional data field to keep track of health if the boxes the fall on the blog cause the different amounts of damage. This class overrides the update method of Observer to update the player correctly.

Squish: The squish class will also extend the GameObjects class. This will be used when a collision happens. Collisions will happen between the player and the boxes.

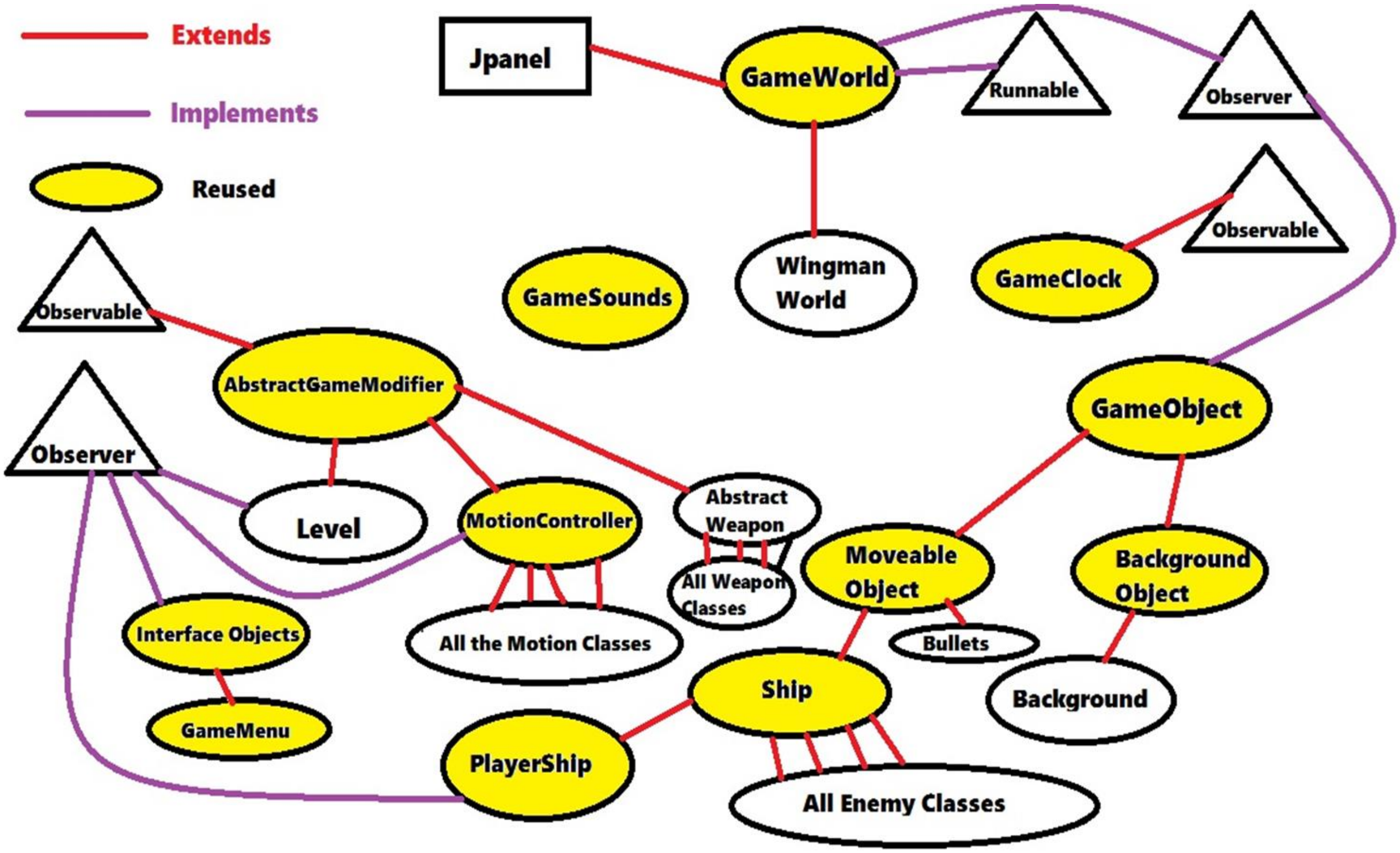
CollisionDetector: This class will be used to check collision between the player and the boxes or the Stop button.
GameEvents: GameEvents implements Observable that will notify the observer which observes it when some defined events happen, there are two defined events: keyboard events and collision events.

GameSounds: It is used to play the music of the game such as background music, and sounds of boxes stacking up or the player getting squished. It should essentially work the same as when implemented for wingman game

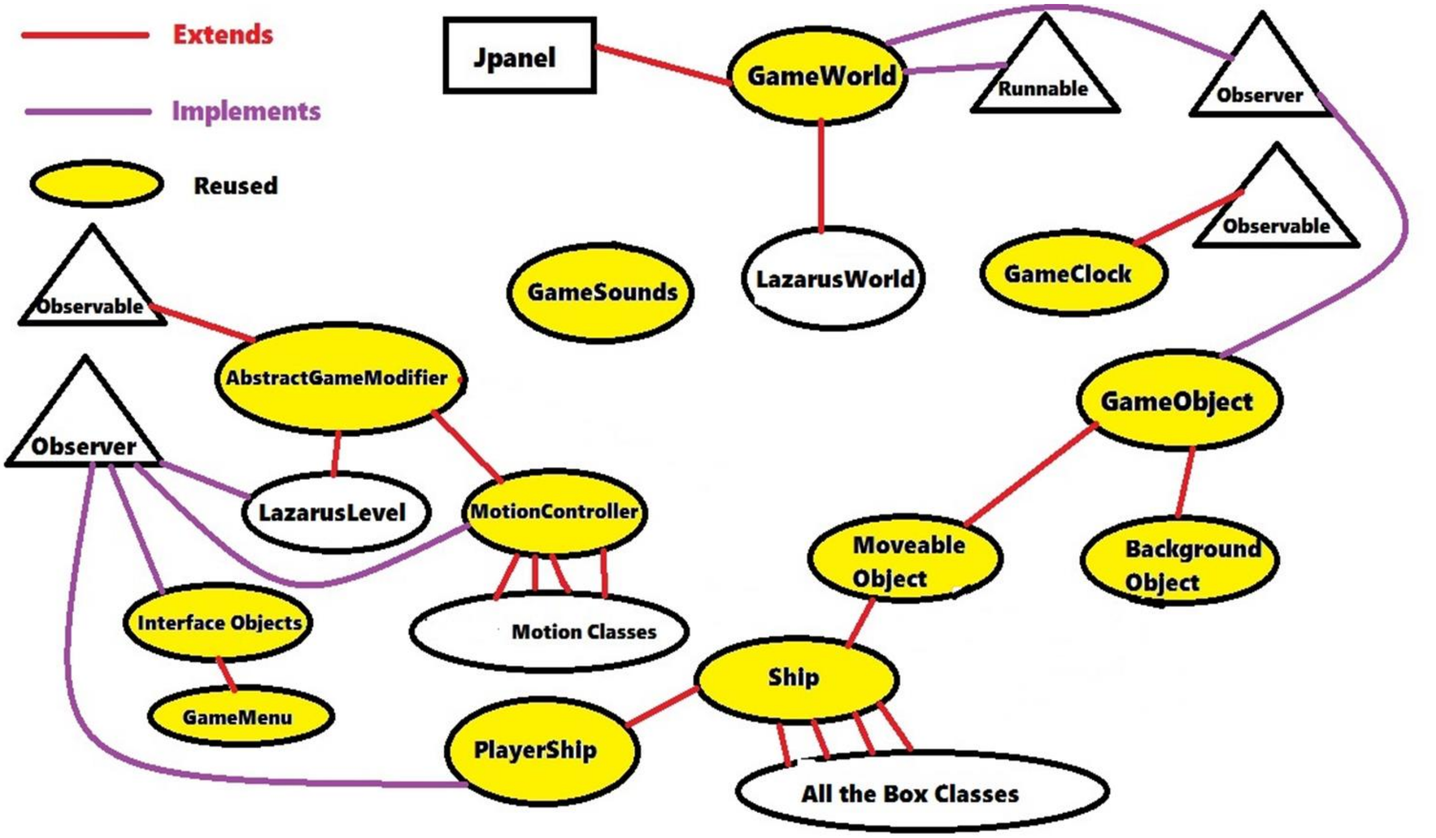
GameWorld: It is used to maintain the static variables that the whole game and other classes will use.

Reusability: Here are the class diagrams for Wingman game and Lazarus game for comparison of classes I reused:

Wingman



Lazarus



Reflection

It really fascinates me that I was able to develop this game with Arthur's code. It taught me that reusability can be very useful and time saving. Just implementing a new game out of someone's existing code is a real hands on experience. First I had to write down what I was going to need for this new game. The process was great learning experience because it taught me that having a plan before coding can come very useful. I then implemented the necessary classes and methods to get the game to work.

In the projects we did throughout semester I learned a lot of fundamental concepts which I will use for the rest of my career. This class taught me that it isn't just about the code, documentation and comments are as equally important, especially if others are going to use my code. It's been a fun semester and thank you for teaching us this course.