Tank Game Documentation

David Lau

Tank Game

The purpose of this assignment is to create an object oriented solution for a two player tank game. For this assignment, I looked through the Wingman code and removed the irrelevant classes relating to the tank game, such as the enemies' ship classes. I then modified the remaining useful classes to run a functional tank game. I used the latest version of NetBeans and JDK to compile and run our tank game. To run the game, I need to have all the tankgame classes/resources in our source code file. To run via cmd/terminal, go to your src file and compile by typing "Javac tankgame/GameWorld.java" then to execute type "Java tankgame.GameWorld". From there, just hit the run project button to execute the game.

When I started this assignment, I assumed that most of the code from the Wingman class could be refused or modified to complete our tank game assignment, but as I continued to develop the tank game, I figured out that the code from the Wingman class wasn't enough, and that I had to do some implementations to the tanks, player's objects, and projectiles on our own. Below, is a brief description of each class and the class diagram.

Class Descriptions (Tank Game):

GameWorld (tankgame): The main class that loads the tankgame, resources, and minimap/player's screen.

GameClock: Gets frame/time

GameSound: Plays sound

Background: extends BackgroundObject; sets background image for map

BackgroundObject: Abstract class for objects that can't move

BigExplosion: Creates explosion object for player's death

Bullet: Extends MoveableObject; sets player's projectile

DestructibleWall: Extends BackgroundObject; sets wall objects in the game that are destructible

GameObject: Abstract class for tankgame's objects

IndestructibleWall: Extends Background Object; sets objects in the game that are indestructible

MoveableObject: Abstract class for objects that can move in the game

PlayerShip: Extends data and methods from Ship class. Player's tank object. Specifies values such as lives, speed, projectile, motion, ect…

Ship: A higher level class for PlayerShip. Extends MoveableObject. Intended to reuse this class for the second game

AbstractGameModifier: Makes changes and updates to GameWorld

Level: Sets the tankgame world with objects from a level.txt file. Each number from the text file represents a game object

InputController: Sets player's controls

MotionController: Abstract class to update object's motions

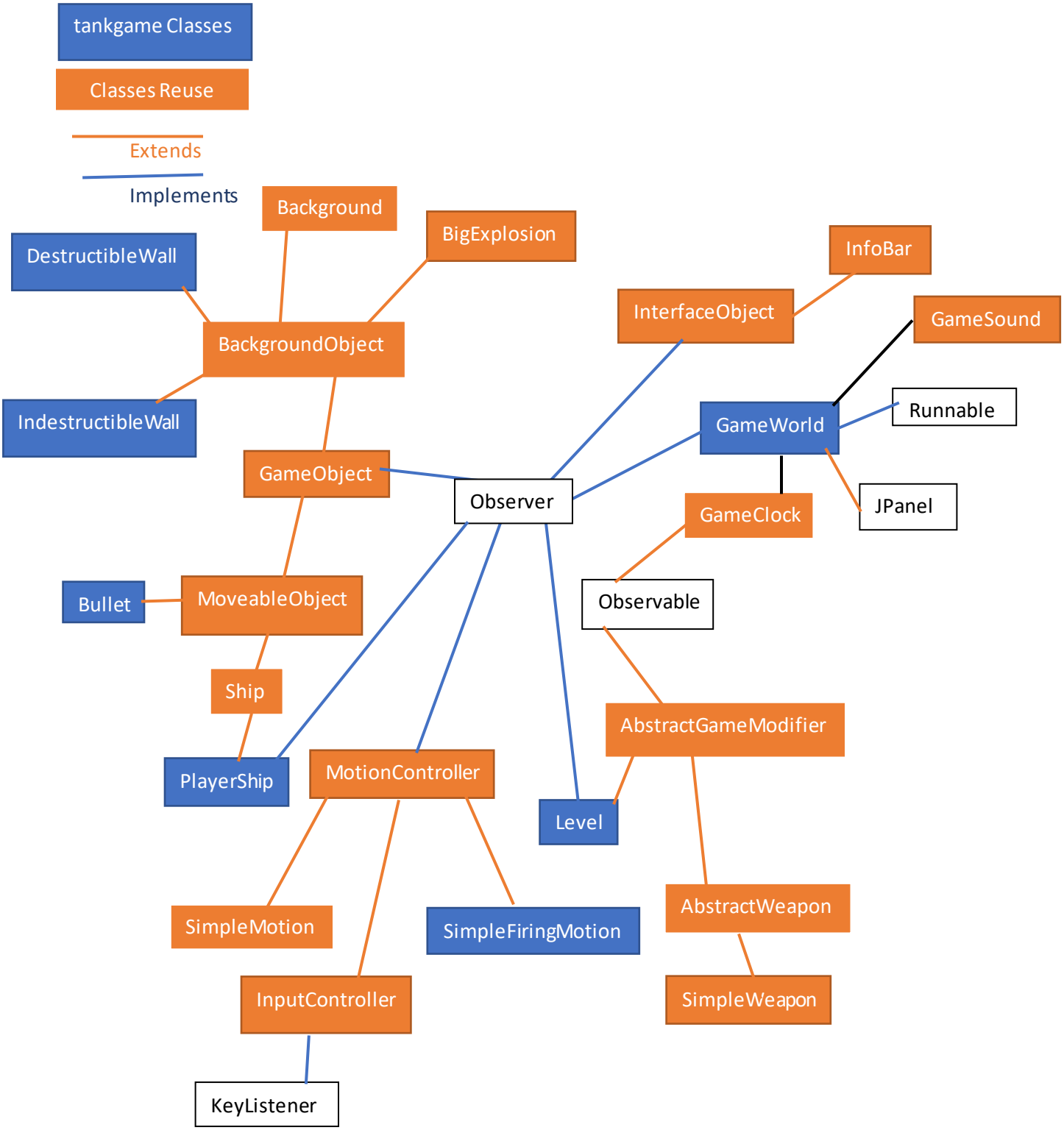
SimpleFiringMotion: Specifies projectile's direction in alignment with player's tank object

SimpleMotion: Moves objects in the game

InfoBar: Updates player's health, score throughout game

InterfaceObject: Abstract class for user interface

Tank Game


tankgame Classes
Classes Reuse
Extends
Implements
Background
BigExplosion
DestructibleWall
InfoBar
InterfaceObject
GameSound
BackgroundObject
IndestructibleWall
GameWorld
Runnable
GameObject
Observer
GameClock
JPanel
Bullet
MoveableObject
Observable
Ship
AbstractGameModifier
PlayerShip
MotionController
Level
SimpleMotion
SimpleFiringMotion
AbstractWeapon
InputController
SimpleWeapon
KeyListener

Conclusions:

In this assignment, I learned the value of object oriented programming. The ability to reuse classes makes programming a lot easier. I also got a better understanding of how inheritance, and encapsulation works. Object oriented programing is a valuable skill for programming and it will be helpful for future assignments. Some challenges I had to overcome was coming up with my own implementation for a child class. For example, I had a higher-level class Ship, but I had to make a PlayerShip so that I could refuse the Ship class for my next game. I had to write my own methods and data that would give the PlayerShip class to function properly as a tank object in the tankgame. I overcame this problem by going back to the higher-level class (Ship) to understand what state and behavior the PlayerShip will be inheriting, and I worked with that information to come up with new methods that made the Player tank function properly. Overall, this assignment has given me the skills of OOP, enhanced my understanding on the concepts of OOP and SOLID, and has helped me transition from a programmer to software developer.