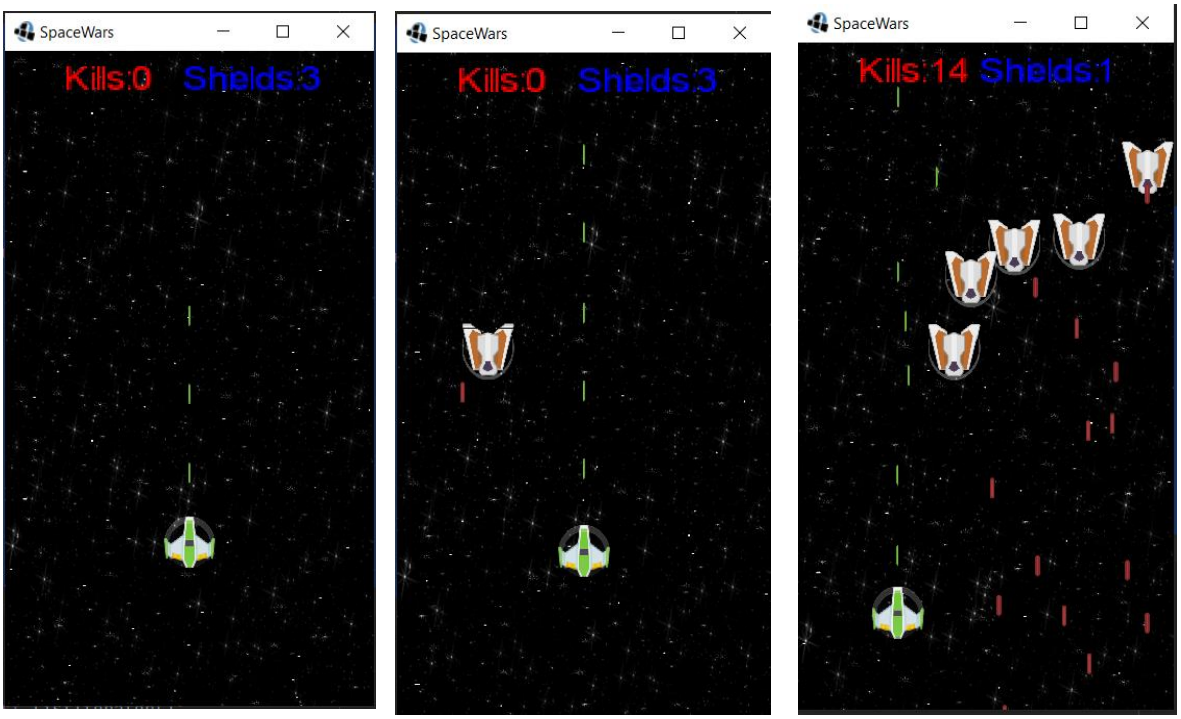
Custom Game Final Version: Space Wars

Emre Serdar
B00773253

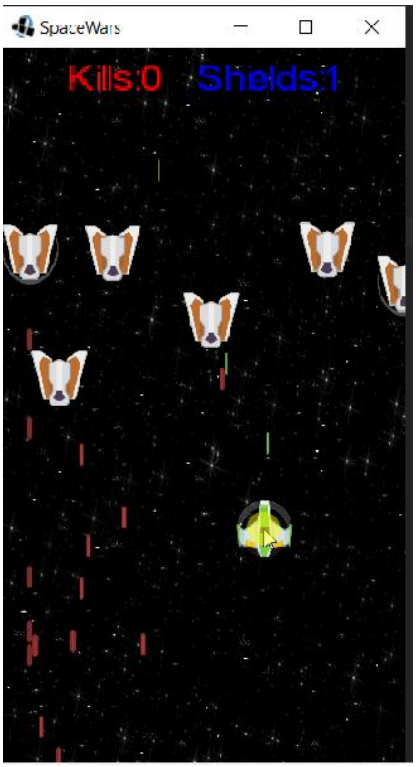
Game Play : https://youtu.be/MMpKBf9QcGE

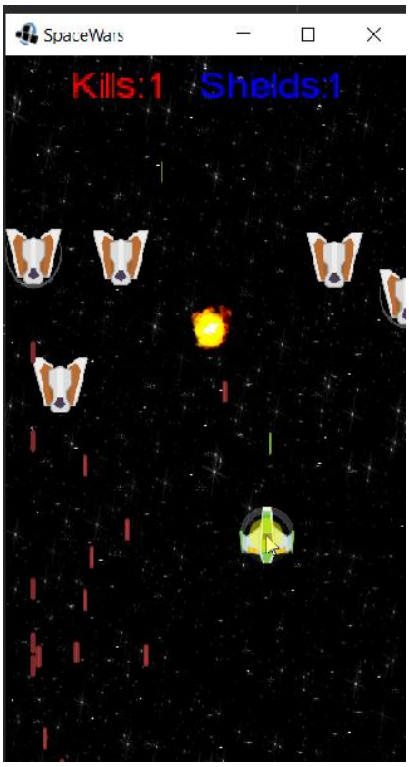


According to game play video and screen shots it can easily be seen that my game includes a player spaceship and enemy spaceships. When the game starts, enemy ships start to spawn on top half screen with 2 seconds as intervals. Both player and enemy ships create laser fires with their own color. Enemy ships move around randomly on top half screen. Player ship is able to move with user click/holds or keyboard inputs (mostly for desktop version of application). The player spaceship has 3 shields at the beginning of game and enemy ships has only 1.

If we look at the ships on Figure 1 and Figure 2, we can see the difference between a ship with shield and a ship without shield. When enemy ship gets 2 laser fires, first one takes down the shield of it and second one kills or destroys the enemy ship. If the player ship gets 3 laser fires (since it has 3 shields at the beginning), they are going to take down player ship's shield. If player ship gets a laser fire when it does not have a shield, player ship is also destroyed, and game closes itself since game is over.

Also, when either enemy ships or player ships are destroyed, there is a small explosion animation occurring on the location of destroyed ship.





When the player ship's laser fire hits an enemy without shield, it explodes the enemy ship and the number of kills getting increased. On the other, if player ship gets a laser fire it's number of shields decreased. I know the designs of fonts (Kills and Shields) are not perfect, however, in my view, the vital parts are the game play itself. Therefore, I could not give much attention to fonts.

Since there are a lot of lines of code and since I have written a lot of comments, I am going to mention parts which I think they are important.

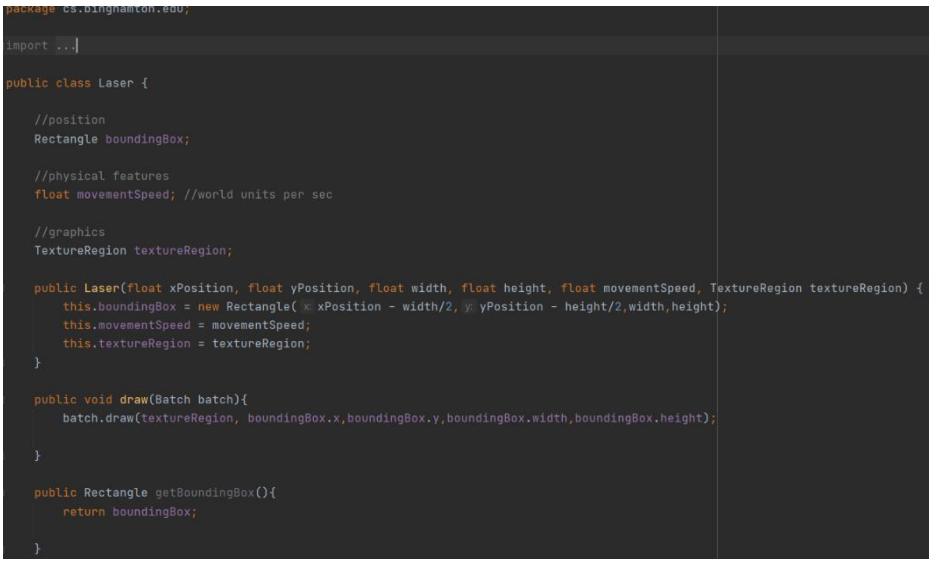
Firstly, while doing this project, I have learnt how to create a background which is moving.

```
@Override
public void render(float delta) {
    batch.begin();

    //scrolling background
    renderBackground(delta);

    //getting keyboard and touch input
    getInput(delta);
    playerShip.update(delta);
```

So, in the render method, my first call is renderBackground() method.

```java
private void renderBackground(float delta) {

    //moving backgrounds with different speeds
    backgroundOffsets[0] += delta * backgroundMaxScrollingSpeed /8 ;
    backgroundOffsets[1] += delta * backgroundMaxScrollingSpeed /4 ;
    backgroundOffsets[2] += delta * backgroundMaxScrollingSpeed /2 ;
    backgroundOffsets[3] += delta * backgroundMaxScrollingSpeed;

    //to avoid printing stuff which cannot be seen from the user
    for (int layer=0; layer<backgroundOffsets.length; layer++){
        if(backgroundOffsets[layer] > WORLD_HEIGHT){
            backgroundOffsets[layer] = 0;
        }
        //drawing layer
        batch.draw(backgrounds[layer], x: 0,
                -backgroundOffsets[layer],
                WORLD_WIDTH,WORLD_HEIGHT);
        batch.draw(backgrounds[layer], x: 0,
                y: -backgroundOffsets[layer] + WORLD_HEIGHT,
                WORLD_WIDTH,WORLD_HEIGHT);

    }

}
```

To make background as moving, I have used the same space background 4 times, however, 3 of them are transparent. Then, I have moved backgrounds with different speeds. Hence, I have had a smooth background which looks like going upside of screen.



```java
package cs.binghamton.edu;

import ...

public class Laser {

    //position
    Rectangle boundingBox;

    //physical features
    float movementSpeed; //world units per sec

    //graphics
    TextureRegion textureRegion;

    public Laser(float xPosition, float yPosition, float width, float height, float movementSpeed, TextureRegion textureRegion) {
        this.boundingBox = new Rectangle( x: xPosition - width/2, y: yPosition - height/2,width,height);
        this.movementSpeed = movementSpeed;
        this.textureRegion = textureRegion;
    }

    public void draw(Batch batch){
        batch.draw(textureRegion, boundingBox.x,boundingBox.y,boundingBox.width,boundingBox.height);

    }

    public Rectangle getBoundingBox(){
        return boundingBox;

    }
```

I have created the Laser class to draw lasers infinitely.

```
//create new lasers
renderLasers(delta);
```

To create new lasers, renderLasers(delta) (which takes an argument deltaTime, the time spent) is called.

```
//set up game objects
playerShip = new PlayerShip( xCenter: WORLD_WIDTH/2,  yCenter: WORLD_HEIGHT/4,
        width: 10, height: 10,
        movementSpeed: 48, shield: 3,
        laserWidth: 0.4f,  laserHeight: 4,  laserMovementSpeed: 30,  timeBetweenShots: 0.5f,
       playerShipTextureRegion, playerShieldTextureRegion, playerLaserTextureRegion);

enemyShipLists = new LinkedList<>();
playerLaserList = new LinkedList<>();
enemyLaserList = new LinkedList<>();
explosionList = new LinkedList<>();
```

First, since each laser fire should remove after touching the ship or after exceeding the boundary of screen, there are a lot of object are getting on screen and out screen. Therefore, to create a laserlist, linked lists are best data structures in my view.

```
416     private void renderLasers(float delta){
417         //player ship lasers
418         if (playerShip.canFireLaser()){
419             Laser[] lasers = playerShip.fireLasers();
420             for (Laser laser: lasers){
421                 playerLaserList.addAll(Arrays.asList(lasers));
422             }
423         }
424
425         //enemy ship lasers
426         ListIterator<EnemyShip> enemyShipListIterator = enemyShipLists.listIterator();
427         while(enemyShipListIterator.hasNext()) {
428             EnemyShip enemyShip = enemyShipListIterator.next();
429
430             if (enemyShip.canFireLaser()) {
431                 Laser[] lasers = enemyShip.fireLasers();
432                 for (Laser laser : lasers) {
433                     enemyLaserList.addAll(Arrays.asList(lasers));
434                 }
435             }
436         }
```

In the render lasers method, first I check if ships can fire laser. canFireLaser is a simple method which returns a boolean according to the equation “timeSinceLastShot – timeBetweenShots >= 0”. To avoid creating infinite number of lasers each time, putting a delay between them make it much smooth. Time between shots are decided while defining the PlayerShip object which extends Ship class.

```
//set up game objects
playerShip = new PlayerShip( xCenter: WORLD_WIDTH/2,  yCenter: WORLD_HEIGHT/4,
        width: 10, height: 10,
        movementSpeed: 48, shield: 3,
        laserWidth: 0.4f,  laserHeight: 4,  laserMovementSpeed: 30,  timeBetweenShots: 0.5f,
       playerShipTextureRegion, playerShieldTextureRegion, playerLaserTextureRegion);
```

```
//enemy ships and movement;
ListIterator<EnemyShip> enemyShipListIterator = enemyShipLists.listIterator();
while (enemyShipListIterator.hasNext()) {
    EnemyShip enemyShip = enemyShipListIterator.next();
    moveEnemy(enemyShip,delta);
    enemyShip.update(delta);
    enemyShip.draw(batch);
}
```

```
private void spawnEnemyShips(float delta){
    enemySpawnTimer+=delta;

    //to spawn enemies on different Location
    float x = SpaceWars.random.nextFloat()*(WORLD_WIDTH+1);

    if (enemySpawnTimer > timeForEnemySpawn){
        enemyShipLists.add(new EnemyShip(x, yCenter: WORLD_HEIGHT*3/4,
                width: 10, height: 10,
                movementSpeed: 30, shield: 1,
                laserWidth: 0.4f, laserHeight: 4, laserMovementSpeed: 30, timeBetweenShots: 0.8f,
                enemyShipTextureRegion, enemyShieldTextureRegion, enemyLaserTextureRegion));

        enemySpawnTimer -= timeForEnemySpawn;
    }
```

```
48      private float timeForEnemySpawn = 2f; //2 seconds
```

```
149
150         //cannot add more enemies if there are already 5
151         if(enemyShipLists.size()<7){
152             spawnEnemyShips(delta);
153         }
```

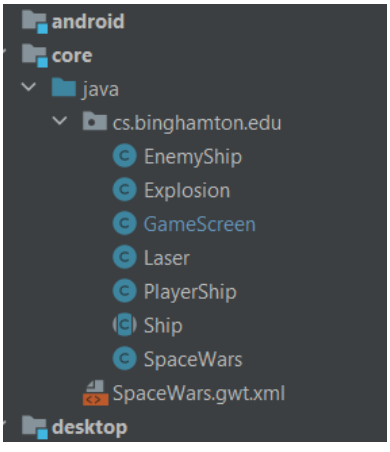
With the spawnEnemeyShips function, it spawns enemies each 2 seconds. To avoid creating a lot of enemies and to make game play better, I have defined a condition. If there are already 7 enemies are already on the screen, another enemy ship cannot be added until one of them is destroyed.

```
private void detectCollision(){
    //for each player laser fire, if it intersects an enemy ship
    ListIterator<Laser> iterator = playerLaserList.listIterator();
    while ( iterator.hasNext()) {
        Laser laser = iterator.next();
        ListIterator<EnemyShip> enemyShipListIterator = enemyShipLists.listIterator();
        while (enemyShipListIterator.hasNext()) {
            EnemyShip enemyShip = enemyShipListIterator.next();

            if (enemyShip.intersects(laser.boundingBox)) {

                //calculate how many shield does the ship has, then if ship can be destroyed
                // reduce the number of shields if there are any
                if(enemyShip.hitAndCheckDestroyed(laser)){
                    enemyShipListIterator.remove(); // removing the destroyed enemy ship
                    explosionList.add(new Explosion(explosionTexture,
                            new Rectangle(enemyShip.boundingBox),  animationTime: 0.7f));
                    kills ++;
                }

                //remove laser
                iterator.remove();
                break;
```

To decide if a player ship's laser fire contact with enemy ship, I have defined a list iterator with laser object. With the while loop, I have checked if tere is a laser in the linked list of lasers, then I have checked if laser hits the enemy ship with the enemyShip.intersect() function. Then, I have created a new explosion texture and increased kills (destroyed ships). Lastly, removed the laser since it hits a ship and break the second while loop since I do not want that while loop goes again an again to fin enemy ship.

```
//to find out if any laser intersects with fire lasers
public boolean intersects(Rectangle rectangle1) { return boundingBox.overlaps(rectangle1); }


public boolean hitAndCheckDestroyed(Laser laser){
    if (shield>0){
        shield--;  //to decrease number of shield if any laser hits ships
        return false; //since there are equal or more than 1 shield
    }
    return true; //ship does not have a shield, so it can be destroyed if laser fire hits

}
```

Intersects and hitAndCheckDestroyed function.



As I said, even though it looks like a simple game, there is a lot going on the backend/background. Therefore, I cannot mention every each line of code instead of I have explained the parts which were important for me.

References:

- https://opengameart.org/content/explosion (for explosion animation)
- https://kenney.nl/assets/space-shooter-redux (for ship, shield, laser textures)