# Space ThunderZ

By GGWP_GibMe15

Members


| | | |
|---|---|---|
| Non | Kositkittiwanit | ID : 5930264521 |
| Gun | Kaewngarm | ID : 5931006121 |

# Space ThunderZ

By GGWP_GibMe15

## 1 Introduction

Space ThunderZ is a vertical scroller shoot-em-up game developed as a term project of 2110215 Programming Methodology class. It is written entirely with Java language with Javafx library. It uses various crucial aspects of Java language such as Object Oriented Programming (OOP), thread, image and audio integration, and many more. For example, inheritance, polymorphism, access modifier setter/getter are used in this project.This project has begun on 17th November 2017, and with dedication and tremendous endeavour, it was finally completed on 11th December 2017. We hope that it's fun to play for both adults and children and the instructors as well.

## 2 User Manual

Space ThunderZ is a single player game which a player plays as a spaceship that needs to shoot the enemies and avoid their bullets as well as avoid colliding with them by moving . The aim is to get the highest scores as many as possible. This is somewhat similar to a shoot-em-up game like Space Invaders and Galaga. In this game, a player health is displayed as a health bar, and will decrease whenever a player takes a hit from an enemy's bullet or collide with an enemy. If the player health bar is depleted completely, the game will be over.

Whenever a player has shot down an enemy (which is indicated by an explosion), he/she will be rewarded with points according to the enemy type that has been destroyed.

The game has three scenes, a main menu scene, a game scene , and a game over scene. The main menu scene is the first one shown up when players start the program.


*Figure 1: The main menu scene of the application*

At the main menu scene, the player can:

- Press ENTER to start the game or
- Press ESC to exit the program.

When a player presses ENTER to start the game. The program's window switches to the game scene as shown in figure 2. There are three aspects displaying on the game scene.

1. Score text at the top-right corner of the screen.
2. A player spaceship at the bottom-center of the screen. A player ship model will be randomized for more challenge because of unpredictability.
3. A health bar at the bottom of the screen. (Player has 1000 Hp)
4. A bonus item status (how long will a bonus item last , and how many missile left) on the top-left corner.


*Figure 2: The game screen of the application.*

- A player can move the ship by pressing arrows keys.
- A player can shoot a laser once by pressing space, or continuously by holding spacebar.
- A player can shoot a missile (if available) by pressing CTRL.

After the game has been started the enemies will start spawning randomly on the top of the screen and move downward, some will also move left and right and shoot as well. The player will have to shoot them for points and to progress further. There are 5 types of normal enemy and 1 boss. The enemy model can be seen in figure 3 below.

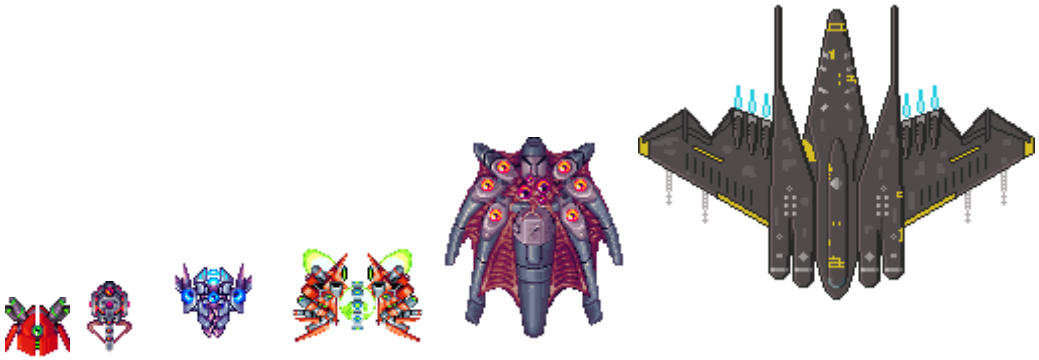

*Figure 3: The enemies.From left to right(bug, squid, eyeball, wing, big, and a boss)*

- Bug move downward only ,rewards 1 point. (Spawn chance 40%)
- Squid move downward only and shoot downward, rewards 3 points. (Spawn chance 30%)
- Eyeball move downward only and rapidly shoot downward, rewards 5 points. (Spawn chance 15%)
- Wing move downward only and rapidly shoot in three directions (like a trident), rewards 10 points. (Spawn chance 10%)
- Big move in elliptical pathrapidly and shoot infive directions, rewards 10 points. (Spawn chance 5%)
- Boss move in horizontal and vertical path and rapidly and shoot huge bullet downward, rewards 30 points. (Spawn chance Scale with the score(*for surprise of course*))

These enemies will spawn randomly and the number of them and the spawn chance will be scaled and adjusted with the amounts of points a player has. Be warned that the game will not be easy as the difficulty is scale with the points that a player has!

If a player takes hit from enemy bullet or collide with an enemy, the health bar will decrease and if it's gone, the player ship will explode and the game will be over.

*Figure 4: The health bar. From left to right(more than 65% health, 65% to 25% health, less than 25% health)*

As the game proceeds, there will be a bonus item spawning randomly st random time. This item will continue to move downward and if the player catches it, it will grant a power up or bonus to the player. The bonus items are Hp box, Missile box, and a Triple Gun box.

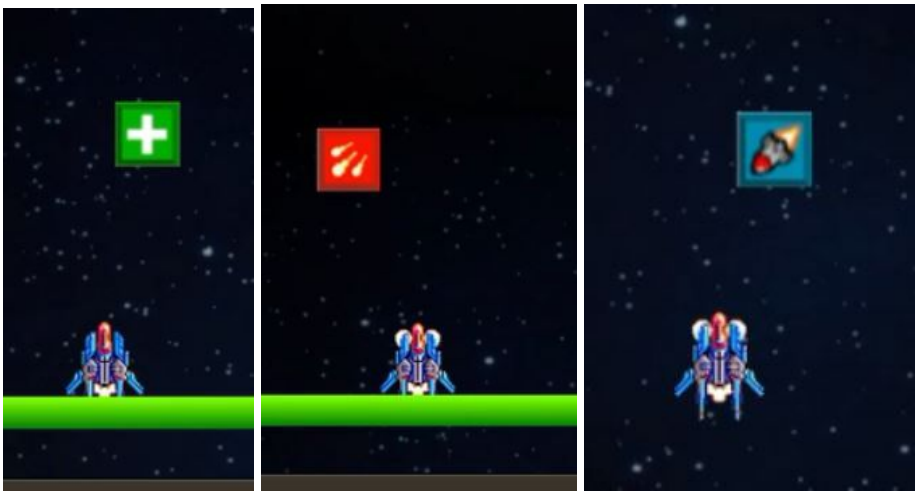

*Figure 5: Thebonus item. From left to right(Hp box, Triple Gun Box, and Missile box)*

These items will grant a bonus as follows

- Hp box will randomly increase a player's health in the range between 50 and 500.

- Triple Gun box will make a playership shoot bullets in three row in front of it. This bonus lasts for 10 seconds. The status of the time left will be indicated at the top-left corner of the screen.

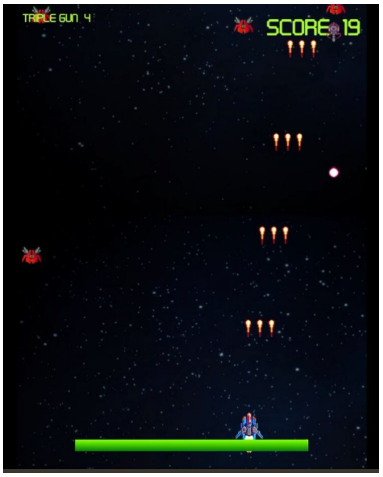


*Figure 6: Triple Gun in action.*

- Missile box will allow a player to shoot a missile by pressing CTRL. The missile can be stored and can be used anytime. The status of how many missile left is displayed on top-left corner of the screen. After the missile has been shot and hit the enemy, it will explode, destroying enemy in a large area.

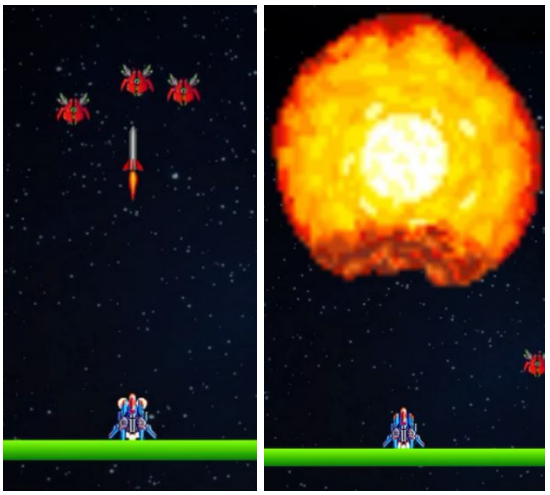

*Figure 7: Missile in action.*

Finally, when the player heath is depleted and the player ship explodes, the game will end and switch to Game Over screen, which displays the overall score.
At the game over menu scene, the player can:

- Press ENTER to restart the game or
- Press ESC to exit the program.

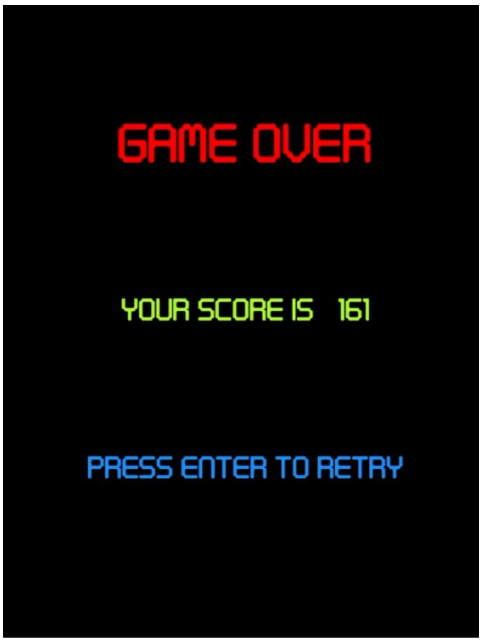


*Figure 8: Game Over screen.*

# 3 Implementation Details

## 3.1 Package application

### 3.1.1 Class Main extends Application

#### 3.1.1.1 Method

| | |
|---|---|
| + void start(Stage primaryStage) | The main entry point for the JavaFX applications. |
| + void main(String[] args) | An entry point of the application. |

## 3.2 Package drawing

### 3.2.1 Class GameOverScreen extends Canvas

#### 3.2.1.1 Field

| | |
|---|---|
| - Font SCORE_FONT | Font used to draw the score. |
| - Font TITLE_FONT | Font used to draw the "GAME OVER" title. |
| - AudioClip music | Music that's played in this screen |

#### 3.2.1.2 Constructor

| | |
|---|---|
| + GameOverScreen() | • Initialize canvas. The canvas width and height is equals to `SceneManager.SCENE_WIDTH` and `SceneManager.SCENE_HEIGHT`.<br>• Play `music`<br>• Set fill color to BLACK, and draw the background with `fillRect.`<br>• Set text alignment to `CENTER`, set fill color to `RED`, set font to `TITLE_FONT`, and draw a fill text "GAME OVER" with the position x and y of `SceneManager.SCENE_WIDTH/2` and `SceneManager.SCENE_HEIGHT/4` respectively.<br>• Then set font to SCORE_FONT, set fill color to `GREENYELLOW`, and draw the fill text score with the position x and y of `SceneManager.SCENE_WIDTH/2` and `SceneManager.SCENE_HEIGHT*2/4` respectively.<br>• Then set fill color to `DODGERBLUE`, and draw the fill text "Press Enter to retry" with the position x and y of `SceneManager.SCENE_WIDTH/2` and `SceneManager.SCENE_HEIGHT*3/4` respectively.<br>• Add Event Handler. |

#### 3.2.1.3 Method

| | |
|---|---|
| - void addKeyEventHandler() | Add a KeyEvent handlers (KeyPressed) to the canvas.<br>• When player press **ENTER**, stop the music and use GameMain to start new game.<br>• When player press **ESC,** exit the game. |

### 3.2.2 Class GameScreen extends Canvas

#### 3.2.2.1 Field

| | |
|---|---|
| - int FPS | Number of frame rates per second. Default is 60. |
| - long LOOP_TIME | Time period between each update of a game animation. |
| - Thread gameAnimation | A thread for game animation. |
| - boolean isAnimationRunning | The flag indicate that the game is start and not end yet. |
| - Queue<Bullet> pendingBullet | The Queue used to store the Bullet that waiting to be added to RenderableHolder List `entities.` |

#### 3.2.2.2 Constructor

| | |
|---|---|
| + GameScreen(double width, double height) | • Initialize canvas. The canvas width and height is equals to the given parameters.<br>• `setVisible` to true.<br>• Set `isAnimationRunning` to false.<br>• Initialize `pendingBullet` as |

| | |
|---|---|
| | `ConcurrentLinkedQueue`.<br>• Add Event Handler. |

#### 3.2.2.3 Method

| | |
|---|---|
| + void startAnimation() | Start the game animation loop thread and set isGameRunning to true. |
| + void stopAnimation() | Set isAnimationRunning to false. |
| - void animationLoop() | The game animation loop:<br>• The loop will stop when isGameRunning is false.<br>• The animation will update when the time has pass by a LOOP_TIME.<br>• For each loop, call updateAnimation(). |
| + void updateAnimation() | Draw the screen interface<br>Request JavaFX Application Thread using Platform.runLater to run the following tasks.<br>• While `pendingBullet` is not empty, poll the Bullet and add them to RenderableHolder instance.<br>• Set fill color to black.<br>• Iterating through RenderableHolder instance entities inside the synchronized block. If entity is visible and not destroyed, call the entity draw(gc) method.<br>• Update the RenderableHolder instance. |
| - void addKeyEventHandler() | Add two KeyEvent handlers to the canvas.<br>• When player press **Arrow Key** or **Spacebar** or **Ctrl** (KeyPressed), use CharacterInput to set pressed KeyCode and pass the `true` parameter.<br>• When player release **Arrow Key** or **Spacebar** or **Ctrl** (KeyReleased), use CharacterInput to set pressed KeyCode and pass the `false` parameter. |
| + void addPendingBullet(Bullet a) | Method used for adding Bullet to the `pendingBullet`. |

### 3.2.3 Class MainMenu extends Canvas

#### 3.2.3.1 Field

| | |
|---|---|
| - Font MENU_FONT | Font used to draw the Menu. |
| - Font TITLE_FONT | Font used to draw the "Space Thunderz" title. |
| - AudioClip music | Music that's played in this screen.<br>Default value is RenderableHolder.mainMenuMusic. |

#### 3.2.1.2 Constructor

| | |
|---|---|
| + MainMenu() | • Initialize canvas. The canvas width and height is |

| | |
|---|---|
| | equals to `SceneManager.SCENE_WIDTH` and `SceneManager.SCENE_HEIGHT`.<br>• Play `music`<br>• Draw the background image from RenderableHolder. Width and height is equals to `SceneManager.SCENE_WIDTH` and `SceneManager.SCENE_HEIGHT`.<br>• Set text alignment to `CENTER`, set fill color to `DODGERBLUE`, set font to `TITLE_FONT`, and draw a fill text "Space Thunderz" with the position x and y of `SceneManager.SCENE_WIDTH/2` and `SceneManager.SCENE_HEIGHT/4` respectively.<br>• Then set font to MENU_FONT, and draw a fill text "Press Enter to start" with the position x and y of `SceneManager.SCENE_WIDTH/2` and `SceneManager.SCENE_HEIGHT*3/4` respectively.<br>• Add Event Handler. |

#### 3.2.1.3 Method

| | |
|---|---|
| - void addKeyEventHandler() | Add a KeyEvent handlers (KeyPressed) to the canvas.<br>• When player press **ENTER**, stop the music and use GameMain to start new game.<br>• When player press **ESC,** exit the game. |

## 3.3 Package game

### 3.3.1 Class GameMain

#### 3.3.1.1 Field

| | |
|---|---|
| - GameScreen gameScreen | A game screen |
| - BackgroundMusic gameBgm | A background music |
| - GameLogic gameLogic | A game logic |
| - GameOverScreen gameOver | A game over screen |

#### 3.3.1.2 Method

| | |
|---|---|
| + void newGame() | Start new game :<br>• Initialize `gameScreen` with width and height of `SceneManager.SCENE_WIDTH` and `SceneManager.SCENE_HEIGHT` respectively.<br>• Initialize `gameBGM`.<br>• Initialize `gameLogic` with `gameScreen` as parameter.<br>• Switch to the game scene by calling `gotoSceneOf` method of SceneManager.<br>• Start `gameLogic`. |

| | |
|---|---|
| | • Start animation in `gameScreen`.<br>• Start BGM in `gameBgm`. |
| + void stopGameLogicAndAnimation() | Stop animation, game logic and BGM by calling the stop method in each class object. |
| + void stopGame() | Stop the game<br>• Call `stopGameLogicAndAnimation()`<br>• Switch to game over scene by calling `displayGameOverResult()`<br>• Clear CharacterInput<br>• Clear RenderableHolder |
| - void displayGameOverResult() | Display the game over and result scene. |

## 3.4 Package input

### 3.4.1 Class CharacterInput

#### 3.4.1.1 Field

| | |
|---|---|
| - ArrayList<KeyCode> keyPressed | A list that holds the input KeyCode(except Ctrl key) that is being press at the time. |
| - boolean isHoldingCtrlKey | The flag that indicate whether the Ctrl key is being pressed/hold and not yet released. |
| - Queue<KeyCode> triggeredCtrl | An instance of ConcurrentLinkedQueue that holds the input Ctrl KeyCode that should be triggered in a next polling time. |

#### 3.4.1.2 Method

| | |
|---|---|
| + boolean getKeyPressed(KeyCode keycode) | Method used to check whether the `keycode` is being pressed at the time. |
| + void setKeyPressed(KeyCode keycode, boolean pressed) | Called when the player presses input.<br>If the input `keycode` is Ctrl key, then use triggered input method<br>• If Ctrl is not being pressed/hold and the input is KeyPressed input(`pressed`==true), then add the `keycode` to `triggeredCtrl` queue and set `isHoldingCtrlKey` to true.<br>• Else, if the input is KeyReleased input(`pressed` ==false), set `isHoldingCtrlKey` to false.<br>else<br>• Add keycode to `keyPressed` list, if the input is KeyPressed input(`pressed`==true) and `keycode` is not already in the `keyPressed` list.<br>• If the input is KeyReleased input(`pressed` ==false), remove the `keycode` from |

| | |
|---|---|
| | `keyPressed` list. |
| + Queue<KeyCode> getTriggeredCtrl() | Return the `triggeredCtrl` queue for Class Player to dequeue Ctrl key. |
| + void clear() | Called when starting/reseting a new game.<br>Clear keyPressed, triggeredCtrl, and set isHoldingCtrlKey to false. |

## 3.5 Package logic

### 3.5.1 Class Background implements IRenderable

#### 3.5.1.1 Field

| | |
|---|---|
| - Image bgImage | A background image |
| - double currentY | The current position in Y axis |
| - int imageWidth | The width of background image |
| - int imageHeight | The height of background image |
| - int screenWidth | The width of the screen<br>Default value is 600. |
| - int screenHeight | The height of the screen<br>Default value is 800. |
| - double scrollSpeed | The background scroll speed<br>Default value is 2.1 |

#### 3.5.1.2 Constructor

| | |
|---|---|
| + Background() | • Set `bgImage` to RenderableHolder background image.<br>• Set `imageWidth` and `imageHeight` to the width and height of `bgImage`. |

#### 3.5.1.3 Method

| | |
|---|---|
| + int getZ() | Called when sorting object instance of IRenderable in RenderableHolder.<br>Return the minimum value of Integer. |
| + void updateBackground() | Update background position |
| + void draw(GraphicsContext gc) | Draw and update the background image |
| + boolean isDestroyed() | Always return false |
| + boolean isVisible() | Always return true |

### 3.5.2 Class Bullet extends CollidableEntity

#### 3.5.2.1 Field

| | |
|---|---|
| - Image bulletSprite; | Image of the bullet |
| - int type; | Type of the bullet:<br>• 0 = Player bullet<br>• 1 = Boss Bullet<br>• 2 = Blue round Bullet<br>• 3 = Yellow round Bullet<br>• 4 = Red round Bullet<br>• 5 = Pink/Purple round Bullet<br>• 6 = Missile<br>• 7 = (Special type) Exploded missile |
| - int zCounter | Counter for z value to be assigned to each object of this class, so that no currently available instance of this class is of the same z value.<br>Default value is -500. |
| - int speedX | Speed in x direction |
| - int speedY | Speed in y direction |
| - boolean exploding | The flag indicate whether the missile is hit/collided and is exploding.<br>Default value is false. |

#### 3.5.2.2 Constructor

| | |
|---|---|
| # Bullet(double x, double y, int speedX, int speedY, int side, int type, CollidableEntity e) | • Call parent Class(CollidableEntity) constructor with `speedY`.<br>• Set `speedX`, `speedY`, `side` and `type` to the given parameters.<br>• Set `height`, `width`, `bulletSprite` and `collideDamage` depend on `type`.<br>• Set `z` to the current value of `zCounter`.<br>• Increment zCounter by 1. If value is more than -300, reset it back to -500.<br>• Calculate the `x` and `y` position from the width and height of the CollidableEntity parameter and `x` and `y` and `side` parameters to center the Bullet in front of CollidableEntity that initialize the object of this class.<br>• Set `visible` to true.<br>• Set `destroyed` to false. |

#### 3.5.2.3 Method

| | |
|---|---|
| + void draw(GraphicsContext gc) | Draw the bullet. |
| + void onCollision(CollidableEntity others) | Called when this object collide with others.<br>● If type is 6, set `exploding` to true and set `visible` to false.<br>● Else if type is 7, do nothing<br>● Else decrease the `hp` by `others collideDamage` value. |
| + void update() | Called by gameLogic update loop.<br>Update parameter of Bullet.<br>● If `type` is 7, set `collideDamage` to 0, set `destroyed` to true and set `visible` to false.<br>● If `exploding` is true,<br>○ Set type to 7<br>○ Calculate new `x` and `y` position<br>○ Increase `width` and `height` to 300.<br>○ Set `collideDamage` to 250.<br>○ Create explosion effect by initialize new Explosion class object and add it to RenderableHolder list.<br>○ Play Explosion sound effect.<br>○ Set `exploding` to false.<br>● If `type` is not 7, update `x` and `y` position with `speedX` and `speedY`.<br>● If `hp` is less than or equal 0 or if the bullet is out of the screen, set `destroyed` to true and set `visible` to false. |
| - boolean isOutOfScreen() | Return true if bullet is out of screen, otherwise return false. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity<br>● Rectangle shape for type 0, 1 and 6.<br>● Otherwise Circle shape. |

### 3.5.3 Class CollidableEntity extends Entity

#### 3.5.3.1 Field

| | |
|---|---|
| # double width | The width in pixel of CollidableEntity |
| # double height | The height in pixel of CollidableEntity |
| # double collideDamage | The damage the others CollidableEntity take when collided with this CollidableEntity |
| + int side | The side of CollidableEntity<br>● -1 is Enemy<br>● 1 is Player |

#### 3.5.3.2 Constructor

| | |
|---|---|
| # CollidableEntity(double hp, double speed) | Call parent class constructor with the given parameters. |

#### 3.5.3.3 Method

| | |
|---|---|
| # boolean collideWith(CollidableEntity other) | Check whether this CollidableEntity collide with other CollidableEntity.<br>● Return true when collided<br>● Return false when not collided<br>● Exception<br>○ Bullet will not hit each other.<br>○ Bullet and Items will not hit each other. |
| + void onCollision(CollidableEntity others) | Called when this object collide with others. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity. |

### 3.5.4 Class EBig extends Enemy

#### 3.5.4.1 Field

| | |
|---|---|
| - int bulletDelayTick | A tick that is used to control shooting. Initially it's 0. |
| - double yOffset | The position offset in y axis.<br>Use for calculate position. |
| - GameLogic gameLogic | A game logic |

#### 3.5.4.2 Constructor

| | |
|---|---|
| + EBig(GameLogic gameLogic) | ● Call parent Class(Enemy) constructor with `hp=270` and `speed=0.5`<br>● Set `height` and `width` with height and width of EBig sprite from RenderableHolder.<br>● Set `yOffset` to 0.<br>● Set `visible` to true.<br>● Set `destroyed` to false.<br>● Set x to make this appear at the center of the screen.<br>● Set y to negative of `height` so this Enemy appear/spawn outside the screen.<br>● Set `collideDamage` to 30.<br>● Set `gameLogic` to the given parameter. |

#### 3.5.4.3 Method

| | |
|---|---|
| + void update() | Called by gameLogic update loop.<br>Update parameter of EBig for new position and bullet shooting.<br>● Increment `yOffset` by `speed`.<br>● Store current time.<br>● Calculate `x` and `y` with `yOffset` and current time. EBig move in spiral(Circle/Oval path that continue to move in y direction).<br>● If this is out of the screen<br>○ Set `visible` to false.<br>○ Set `destroyed` to true.<br>● If `bulletDelayTick` complete an interval, shoot 4 type 4 bullets(Red round Bullet) and play laser sound effect.<br>● Increment `bulletDelayTick` by 1. |
| + void draw(GraphicsContext gc) | Draw this EBig sprite. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity.<br>● Circle hitbox. |

### 3.5.5 Class EBoss extends Enemy

#### 3.5.5.1 Field

| | |
|---|---|
| - int originalHp | An original HP of EBoss.<br>Used for drawing HP bar. |
| - GameLogic gameLogic | A game logic |
| - int bulletDelayTick | A tick that is used to control shooting. Initially it's 0. |

#### 3.5.5.2 Constructor

| | |
|---|---|
| + EBoss(GameLogic gameLogic) | ● Call parent Class(Enemy) constructor with `hp=1000` and `speed=0.2`<br>● Set `height` and `width` with height and width of EBoss sprite from RenderableHolder.<br>● Set `visible` to true.<br>● Set `destroyed` to false.<br>● Set x to make this appear at the center of the screen.<br>● Set y to negative of `height` so this Enemy appear/spawn outside the screen.<br>● Set `collideDamage` to 50.<br>● Set `gameLogic` to the given parameter.<br>● Set GameLogic isBossAlive to true. |

#### 3.5.5.3 Method

| | |
|---|---|
| + void update() | Called by gameLogic update loop.<br>Update parameter of EBoss for new position and bullet shooting.<br>● Store current time.<br>● Calculate `x` with current time. EBoss move by swing back and forth between the edge of the screen in x direction.<br>● Increase `y` by `speed` if y is less than 40.(EBoss move down then stop at a certain y position)<br>● If this is out of the screen<br>○ Set `visible` to false.<br>○ Set `destroyed` to true.<br>● If `bulletDelayTick` complete an interval, shoot 1 type 1 bullet (Boss Bullet) and play laser sound effect.<br>● Increment `bulletDelayTick` by 1. |
| + void draw(GraphicsContext gc) | Draw this EBoss sprite and HP bar. |
| - drawHpBar(GraphicsContext gc) | Draw HP bar. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity.<br>● Circle hitbox. |

### 3.5.6 Class EBug extends Enemy

#### 3.5.6.1 Constructor

| | |
|---|---|
| + EBug(double x) | ● Call parent Class(Enemy) constructor with `hp=100` and `speed=2`<br>● Set `height` and `width` with height and width of EBug sprite from RenderableHolder.<br>● Set `visible` to true.<br>● Set `destroyed` to false.<br>● Set x to the given parameter..<br>● Set y to negative of `height` and a random value between 0 to 500 so this Enemy appear/spawn outside the screen.<br>● Set `collideDamage` to 10. |

#### 3.5.6.2 Method

| | |
|---|---|
| + void draw(GraphicsContext gc) | Draw this EBug sprite. |
| + void update() | Called by gameLogic update loop.<br>Update parameter of EBug for new position.<br>● Increase `y` by `speed`.<br>● If this is out of the screen<br>○ Set `visible` to false.<br>○ Set `destroyed` to true. |

| | |
|---|---|
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity.<br>● Rectangle hitbox. |

### 3.5.7 Class EEyeball extends Enemy

#### 3.5.7.1 Field

| | |
|---|---|
| - int bulletDelayTick | A tick that is used to control shooting. Initially it's 0. |
| - GameLogic gameLogic | A game logic. |

#### 3.5.7.2 Constructor

| | |
|---|---|
| EEyeball(GameLogic gameLogic, double x) | ● Call parent Class(Enemy) constructor with `hp=120` and `speed=1`<br>● Set `height` and `width` with height and width of EEyeball sprite from RenderableHolder.<br>● Set `visible` to true.<br>● Set `destroyed` to false.<br>● Set x to the given parameter.<br>● Set y to negative of `height` so this Enemy appear/spawn outside the screen.<br>● Set `collideDamage` to 10.<br>● Set `gameLogic` to the given parameter. |

#### 3.5.7.3 Method

| | |
|---|---|
| + void draw(GraphicsContext gc) | Draw this EEyeball sprite. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity.<br>● Rectangle hitbox. |
| + void update() | Called by gameLogic update loop.<br>Update parameter of EEyeball for new position.<br>● Increase `y` by `speed`.<br>● If this is out of the screen<br>○ Set `visible` to false.<br>○ Set `destroyed` to true.<br>● If `bulletDelayTick` complete an interval, shoot 1 type 2 bullet (Blue round Bullet) and play laser sound effect.<br>● Increment `bulletDelayTick` by 1. |

### 3.5.8 Class Enemy extends CollidableEntity

#### 3.5.8.1 Field

| | |
|---|---|
| - int zCounter | Counter for z value to be assigned to each object of this class, so that no currently available instance of this class is of the same z value.<br>Default value is -200. |

#### 3.5.8.2 Constructor

| | |
|---|---|
| + Enemy(double hp, double speed) | • Call parent Class(CollidableEntity) constructor with the given parameters.<br>• Set `side` to -1.<br>• Set `z` to the current value of `zCounter`.<br>• Increment zCounter by 1. If value is more than -100, reset it back to -200.<br>• Increment GameLogic `currentEnemyNum` by 1. |

#### 3.5.8.3 Method

| | |
|---|---|
| + void onCollision(CollidableEntity others) | Called when this object collide with others.<br>• Decrease the `hp` by `others collideDamage` value.<br>• If hp is less than or equal 0,<br>  ○ If this isn't destroyed yet,<br>  ■ Calculate score<br>  ■ Decrement GameLogic currentEnemyNum by 1.<br>  ■ Initialize explosion effect with Explosion class.<br>  ■ Play explosion sound effect.<br>  ■ Add instance of Explosion to RenderableHolder.<br>  ○ Set `visible` to false.<br>  ○ Set `destroyed` to true. |
| + boolean isOutOfScreen() | Check whether this Enemy instance has gone out of the screen.<br>• Decrement GameLogic `currentEnemyNum` by 1 and return true if this has gone out of the screen.<br>• Otherwise return false. |
| - void calculateScore(Enemy e) | • Calculate the gained score and add them to Score `score`.<br>• If `e` is EBoss, then also set GameLogic `isBossAlive` to false. |

### 3.5.9 Class Entity implements IRenderable

#### 3.5.9.1 Field

| | |
|---|---|
| # double x | The position in x axis. |
| # double y | The position in y axis. |

| | |
|---|---|
| # double hp | The health point |
| # double speed | The speed |
| # int z | The value used for sorting in RenderableHolder to determine what is being draw first and last. |
| # boolean visible | The flag indicate whether this Entity is visible or not. Used by animation loop. |
| # boolean destroyed | The flag indicate whether this Entity is destroyed or not. |

#### 3.5.9.2 Constructor

| | |
|---|---|
| # Entity() | Default constructor<br>• Set `visible` to true.<br>• Set `destroyed` to false.<br>• Set `hp` to 100.<br>• Set `speed` to 20. |
| # Entity(double hp, double speed) | • Set `visible` to true.<br>• Set `destroyed` to false.<br>• Set `hp` to the given parameter.<br>• Set `speed` to the given parameter. |

#### 3.5.9.3 Method

| | |
|---|---|
| + boolean isDestroyed() | Getter for `destroyed` |
| + boolean isVisible() | Getter for `visible` |
| + int getZ() | Getter for `z` |
| + void update() | Called by gameLogic update loop.<br>Used for updating parameter of Entity. |

### 3.5.10 Class ESquid extends Enemy

#### 3.5.10.1 Field

| | |
|---|---|
| - int bulletDelayTick | A tick that is used to control shooting. Initially it's 0. |
| - GameLogic gameLogic | A game logic. |

#### 3.5.10.2 Constructor

| | |
|---|---|
| + ESquid(GameLogic gameLogic, double x) | • Call parent Class(Enemy) constructor with `hp=70` and `speed=2.5`<br>• Set `height` and `width` with height and width |

| | |
|---|---|
| | of ESquid sprite from RenderableHolder.<br>● Set `visible` to true.<br>● Set `destroyed` to false.<br>● Set x to the given parameter.<br>● Set y to negative of `height` so this Enemy appear/spawn outside the screen.<br>● Set `collideDamage` to 10.<br>● Set `gameLogic` to the given parameter. |

#### 3.5.10.3 Method

| | |
|---|---|
| + draw(GraphicsContext gc) | Draw this ESquid sprite. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity.<br>● Rectangle hitbox. |
| + void update() | Called by gameLogic update loop.<br>Update parameter of ESquid for new position and shoot the bullet.<br>● Increase `y` by `speed`.<br>● If this is out of the screen<br>○ Set `visible` to false.<br>○ Set `destroyed` to true.<br>● If `bulletDelayTick` complete an interval, shoot 1 type 5 bullet (Pink/Purple round Bullet) and play laser sound effect.<br>● Increment `bulletDelayTick` by 1. |

### 3.5.11 Class EWing extends Enemy

#### 3.5.11.1 Field

| | |
|---|---|
| - int bulletDelayTick | A tick that is used to control shooting. Initially it's 0. |
| - GameLogic gameLogic | A game logic. |

#### 3.5.11.2 Constructor

| | |
|---|---|
| + EWing(GameLogic gameLogic, double x) | ● Call parent Class(Enemy) constructor with `hp=150` and `speed=0.5`<br>● Set `height` and `width` with height and width of EWing sprite from RenderableHolder.<br>● Set `visible` to true.<br>● Set `destroyed` to false.<br>● Set x to the given parameter.<br>● Set y to negative of `height` so this Enemy appear/spawn outside the screen.<br>● Set `collideDamage` to 10.<br>● Set `gameLogic` to the given parameter. |

#### 3.5.11.3 Method

| | |
|---|---|
| + draw(GraphicsContext gc) | Draw this EWing sprite. |
| + Shape getBoundary() | Called by collideWith(CollidableEntity other)<br>Return the hitbox of this CollidableEntity.<br>● Rectangle hitbox. |
| + void update() | Called by gameLogic update loop.<br>Update parameter of EWing for new position and shoot the bullet.<br>● Increase `y` by `speed`.<br>● If this is out of the screen<br>○ Set `visible` to false.<br>○ Set `destroyed` to true.<br>● If `bulletDelayTick` complete an interval, shoot 3 type 3 bullets (Yellow round Bullet) and play laser sound effect.<br>● Increment `bulletDelayTick` by 1. |

### 3.5.12 Class Explosion implements IRenderable

#### 3.5.12.1 Field

| | |
|---|---|
| - Image explosion[] | An array that store the explosion effect image sequence. |
| - AudioClip sound | An explosion sound effect |
| - int z | The value used for sorting in RenderableHolder to determine what is being draw first and last. |
| - double x | The position in x axis. |
| - double y | The position in y axis. |
| - double width | The width in pixel. |
| - double height | The height in pixel. |
| - int explosionTick | The tick used for drawing the explosion effect image sequence. |
| - boolean destroyed | The flag indicate whether this is destroyed or not. |
| - boolean visible | The flag indicate whether this is visible or not.<br>Used by animation loop. |

#### 3.5.12.2 Constructor

| | |
|---|---|
| + Explosion(double posx, double posy, double width, double height, int originalZ) | ● Set `explosion` to `exploArr` from RenderableHolder. |

| | |
|---|---|
| | • Set `x` and `y` to the given parameters(`posx`,`posy`).<br>• Set `width` and `height` to the given parameters.<br>• Initialize Class Random instance.<br>• Use Random instance to randomly set `sound` to one of the sound effect in `explosions` array from RenderableHolder.<br>• Set `visible` to true.<br>• Set `destroyed` to false. |

#### 3.5.12.3 Method

| | |
|---|---|
| + int getZ() | Getter for `z`. |
| + void draw(GraphicsContext gc) | Draw the explosion effect image sequence and update the explosion effect. |
| + boolean isDestroyed() | Getter for `destroyed`. |
| + boolean isVisible() | Getter for `visible`. |
| + void updateExplosion() | Update the explosion effect.<br>• Increment `explosionTick` by 1.<br>• If `explosionTick` reach 24,<br>  ○ Set `visible` to false.<br>  ○ Set `destroyed` to true. |
| + void playSfx() | Play the explosion `sound` effect. |

### 3.5.13 Class GameLogic

#### 3.5.13.1 Field

| | |
|---|---|
| -Queue<Bullet> pendingBullet | A queue that cotains every bullet that has just been shot. |
| -List<Entity> gameObjectContainer | A list containg every Object that is currently in the game. |
| -int FPS | A rate of frames per second of this game. It is 60. |
| -long LOOP_TIME | A time counter indicate when to update the game.<br>• it is 1000000000 / `FPS` |
| -int gameOverCountdown | A time that is used to delay whenever a player loses. |
| -int maxEnemyCap | Maximun number of enemies on the screen. |
| +int currentEnemyNum | Current number of enemies on the screen. |
| +boolean isBossAlive | A boolean that indicates whether the Boss |

| | |
|---|---|
| | `(EBoss)` is alive or not. |
| -int stageLevel | A number that is use for adjusting game's difficulty. |
| -long nextItemsSpawnTime | A time that the next bonus item will spawn. |
| -GameScreen canvas | A canvas that will be drawn on. |
| -boolean isGameRunning | Indicate whether the game is running or not. |
| -Player player | A current player ship of this game. |
| -EBig ebig | An EBig enemy type. |
| -EBoss eboss | A boss of this game. |
| -EBug ebug | An EBug enemy type. |

#### 3.5.13.2 Constructor

| | |
|---|---|
| +GameLogic(GameScreen canvas) | Initilize the game's logic part.<br>● Initialize new `List` for `gameObjectContainer`.<br>● Set `maxEnemyCap` to 5.<br>● Set `currentEnemyNum` to 0.<br>● Set `stageLevel` to 1.<br>● Set `isBossAlive` to false.<br>● Add new `Background` and `Score` to `RenderableHolder`.<br>● Initilize new `player`.<br>● Spawn enemy using `spawnEnemy`();<br>● Set `this.canvas` to `canvas`.<br>● Set `nextItemSpawnTime` to current nano time + random time from [50 secs to 150 secs).<br>● Initialized new `Queue` for `pendingBullet`. |

#### 3.5.13.3 Method

| | |
|---|---|
| #void addNewObject(Entity entity) | Add new Entity to the container list.<br>● Add entity to gameObjectContainer.<br>● Add entity to RenderableHolder. |
| +void startGame() | Start the game.<br>● Set isGameRunning to true.<br>● Initilize new Thread that runs gameLoop method. |
| +void stopGame() | Stop the game.<br>● Set isGameRunning to false<br>● Clear gameObjectContainer. |

| | |
|---|---|
| | ● Clear pendingBullet. |
| -void gameLoop() | Loop the game.<br>● While this game is still running.<br>○ if elapsed time >= LOOP_TIME<br>■ update the game (updateGame() )<br>○ sleep the Thread for 1 milliseconds. |
| -void updateGame() | Update the current state of the game.<br>● Spawn enemy.<br>● Spawn items.<br>● All all the `Bullet` in `pendingBullet` to `gameObjectContainer`.<br>● Update every `Entity` in `gameObjectContainer`.<br>● Check every object in `gameObjecctContainer` whether they collide with other different `CollidableEntity` or not<br>○ if collide then call `onCollision` method of that object.<br>● Check every object in `gameObjectContainer` whether it is destroyed or not.<br>○ if it's destroyed then remove it.<br>● Check whether `player` is destroyed or not.<br>○ if it's destroyed then decrease `gameOverCountdown` by 1.<br>● Check whether if `gameOverContdown` is 0 or not.<br>○ if `0` then stop the game (`GameMain.stopGame()` ). |
| +void addPendingBullet(Bullet a) | Add a `Bullet` to `pendingBullet` queue.<br>● Add a to `pendingBullet`.<br>● Add a to `canvas`. |
| -void spawnEnemy() | Spawn a new enemy.<br>● Set `maxEnemyCap` to 5 + `stageLevel`.<br>● If game `score` is between 100 * `stageLevel` * 1.5 and 100 * `stageLevel` * 1.5 + 30, and `score` != 0 , and boss is not alive.<br>○ Add new `EBoss` to `gameObjectContainer`.<br>● If `score` >= 100 * `stageLevel` * 1.5<br>○ increase `stageLevel` by 1.<br>● If `currentEnemyNum` < `maxEnemyCap`, random a chance to add an enemy to `gameObjectContainer`.<br>○ 40% add `EBug`.<br>○ 30% add `ESquid`.<br>○ 15% add `EEyeball`.<br>○ 10% add `EWing`. |

| | |
|---|---|
| | ○ 5% add `EBig`. |
| -void spawnItems() | Spawn a new bonus item.<br>● If `nextItemsSpawnTime` <= current time.<br>○ Set `nextItemsSpawnTime` to current time + random time between [50 seconds, 150 seconds).<br>○ Add new bonus item by using chance as a factor.<br>■ 32.5% add `TripleGunBox.`<br>■ 22.5% add `MissileBox`.<br>■ 45% add `HPBox`. |

### 3.5.14 Class HPBox extends Items

#### 3.5.14.1 Field

| | |
|---|---|
| -double HPStorage | The amount of hp that this box will increase the player's health. |

#### 3.5.14.2 Constructor

| | |
|---|---|
| +HPBox(double x) | Set size and location of HPBox and set the amount of `HPStorage`.<br>● Call parent calss constructor (`Items`) with random speed from 1 to 5 as a parameter.<br>● set `HPStorage` with random number in the interval of [50, 500).<br>● set `width` and `height` to those of `RenderableHolder.hpBox`.<br>● set `visbible` to true.<br>● set `destroyed` to false.<br>● set `this.x` to x.<br>● set `y` to `-height -(random nuber from [0,500) ).`<br>● set `collideDamage` to `0`. |

#### 3.5.14.3 Method

| | |
|---|---|
| +void draw(GraphicsContext gc) | Draw `HPBox` onto the screen.<br>● draw `RenderableHolder.hpBox` at x, y position. |
| +void onCollision(CollidableEntity others) | Update this object when collide with other `CollidableEntity` object.<br>● set hp to 0.<br>● set destroyed to true. |

| | |
|---|---|
| | • set visible to false. |
| #double getHPStorage() | A getter for `HPStorage` value. |

### 3.5.15 Class Items extends CollidableEntity

#### 3.5.15.1 Field

| | |
|---|---|
| -int zConter | A z value of an item. Initially it is -725. |

#### 3.5.15.2 Constructor

| | |
|---|---|
| #Items(double speed) | Initilaize the item.<br>• call parent constructor (CollidableEntity)<br>• set side = -1<br>• set z = zCounter<br>• plus zCounter by 1<br>• if (zCounter > -725)<br>  ○ minus zCounter by -750 |

#### 3.5.15.3 Method

| | |
|---|---|
| +Shape getBoundary() | Return the boundary of this object in a Shape type.<br>• Set the Shape as a rectangle Shape start at x, y and with this object size. |
| +void update() | • Add speed to y position.<br>• If this object is out of screen (call isOutOfScreen() )<br>  ○ set visible to false.<br>  ○ set destroyed to true. |
| +boolean isOutOfScreen() | Return true if this object is out of screen and false if otherwise.<br>• If ((int) this.y > SceneManager.SCENE_HEIGHT)<br>  ○ return true.<br>• Else return false. |

### 3.5.16 Class MissileBox extends Items

#### 3.5.16.1 Constructor

| | |
|---|---|
| +MissileBox(double x) | Set size and location of TripleGun box<br>• Call parent calss constructor (`Items`) with random speed from 1 to 5 as a parameter. |

| | |
|---|---|
| | • Set `width` and `height` to those of `RenderableHolder.missileBox`<br>• Set `visbible` to true.<br>• Set `destroyed` to false.<br>• Set `this.x` to x.<br>• Set `y` to `-height -(random nuber from [0,500))`.<br>• Set `collideDamage` to 0. |

#### 3.5.16.2 Method

| | |
|---|---|
| +void draw(GraphicsContext gc) | Draw MissileBox onto screen at x ,y. |
| +void onCollision(CollidableEntity others) | Update this object when collide with other.<br>• Set `hp` to 0.<br>• Set `destroyed` to `true`.<br>• Set `visible` to `false`. |

### 3.5.17 Class Player extends CollidableEntity implements IRenderable

#### 3.5.17.1 Field

| | |
|---|---|
| -Image playerImage | Image of a player ship. At first it is null. |
| GameLogic gameLogic | A GameLogic class. |
| -int bulletDelayTick | A tick that is used to control shooting. Initially it's 0. |
| -int prevbulletTick | A tick that contains last time that a bullet came out. Initially it's 0. |
| -double originalHp | An original hp of a player ship. |
| -long TripleGunTimeOut | A tick that is use for controling how long does the TripleGun bonus last. Initially it's 0. |
| -int missile | The amount of missile that a player has left. Initially it's 0. |
| -int gunMode | The mode of a player gun. Initialy it's 0.<br>• 0 is for normal gun.<br>• 1 is for TripleGun. |

#### 3.5.17.2 Constructor

| | |
|---|---|
| +Player(GameLogic gameLogic) | Initilize a player ship.<br>• Call parent constructor (`CollidableEntity`) with 1000 hp and 20 speed.<br>• Set `playerImage` as a random image in |

| | |
|---|---|
| | `RenderableHolder.playerShip` by acessing array randomly from positio 0 to 5.<br>● Set `this.gameLogic` = gameLogic.<br>● Set `width` and `height` as those of `playerImage`.<br>● set x as `SceneManager.SCENE_WIDTH / 2 - this.width / 2`<br>● set y as `(SceneManager.SCENE_HEIGHT - this.height) - 60`<br>● Set `speed` as 3.<br>● Set `side` as 1.<br>● Set `collideDamage` as 10. |

### 3.5.17.3 Method

| | |
|---|---|
| -void drawHpBar(GraphicsContext gc) | Draw Health bar at the bottom of the screen<br>● hp >= 65% paint Hp bar with Green.<br>● 25%<= hp < 65% paint with yellow.<br>● hp < 25 paint with red.<br>● draw using `fillRect` at `(SceneManager.SCENE_WIDTH / 2 - 200 * percentHp, 750)` with `width and height of (2 * 200 * percentHp, 20)` |
| -drawItemsStatus(GraphicsContext gc) | Draw item status at the top-left corner of the screen.<br>● `setFont` as `RenderableHolder.inGameFontSmall` |
| +void draw(GraphicsContext gc) | Draw the player ship,hp bar, and item status on the screen.<br>● Draw playerImage at x ,y<br>● Call drawHpBar(gc)<br>● Call drawItemsStatus(gc) |

| | |
|---|---|
| +void update() | Update the location of `player` and shoot the bullet if press `SPACE`.<br>Check input by using `CharacterInput.getKeyPressed(KeyCode)`<br>● If is pressing UP key, call `forward(true)`.<br>● If is pressing DOWN key, call `forward(false)`.<br>● If is pressing RIGHTkey, call `turn(true).`<br>● If is pressing LEFTkey, call `turn(false).`<br>● If Triggered CTRL key<br>○ `Add new Bullet to pendingBullet` of `gameLogic` with position x ,y<br>○ speedX = 0, speedY = 30<br>○ side = 1 , type = 6<br>○ and CollidableEntity type = this<br>● If is pressing SPACE<br>○ if `bulletDelayTick - prevbulletTick` > 7<br>■ if `gunMode` is 0 ,then add 1 new `Bullet` to `pendingBullet` of `gameLogic` with this parameter.<br>● (x, y, 0, 20, 1, 0, this)<br>■ if gunMode is 1 ,then add 3 new `Bullet` to `pendingBullet` of `gameLogic` with these parameters.<br>● (x, y, 0, 20, 1, 0, this)<br>● (x - 20, y, 0, 20, 1, 0, this)<br>● (x + 20, y, 0, 20, 1, 0, this)<br>○ `prevbulletTick = bulletDelayTick`<br>● Increase `bulletDelayTick` by 1.<br>● If `TripleGunTimeOut` <= current time.<br>○ Set `TripleGunTimeOut` to 0<br>○ Set gunMode to 0 |
| +void onCollision(CollidableEntity others) | ● Minus `hp` by `others.collideDamage`<br>● If `others` is `HPBox`<br>○ Add this `hp` by `getHpStorage` of others<br>● If `others` is `TripleGunBox`<br>○ Set `gunMode` = 1<br>○ Set `TripleGunTimeOut` = 10 secs<br>● If `others` is `MissileBox`<br>○ Add `missile` by 1 |

| | |
|---|---|
| | • If this `hp` <= 0<br>  ○ Set `destroyed` to true<br>  ○ Set `visible` to false<br>  ○ Add new `Explosion` at x ,y with this `width`, `height` ,and z. Add it to `RenderableHolder` |
| -void forward(boolean b) | Update y position of ship.<br>• If b is `true` and if this ship is not out of screen then move up.<br>• If b is `false` and if this ship is not out of screen then move down. |
| -void turn(boolean b) | Update x position of ship.<br>• If b is `true` and if this ship is not out of screen then move right.<br>• If b is `false` and if this ship is not out of screen then move left. |
| +Shape getBoundary() | Called when checking whether this entity collide with other entity. |

### 3.5.18 Class Score implements IRenderable

#### 3.5.18.1 Field

| | |
|---|---|
| +int score | A field to store current score. |

#### 3.5.18.2 Constructor

| | |
|---|---|
| +Score() | • set `score` to 0. |

#### 3.5.18.3 Method

| | |
|---|---|
| +int getZ() | • Return z value as `Integer.MAX_VALUE` . |
| +draw(GraphicsContext gc) | Draw the score text on the screen. |
| boolean isDestroyed() | Return whether this object is destroyed or not.<br>• Always return false. |
| boolean isVisible() | Return whether this object is visible or not.<br>• Always return true. |

### 3.5.19 Class TripleGunBox extends Items

#### 3.5.19.1 Constructor

| | |
|---|---|
| + TripleGunBox(double x) | Set size and location of TripleGun box<br>• Call parent calss constructor (`Items`) with random speed from 1 to 5 as a parameter.<br>• Set `width` and `height` to those of `RenderableHolder.tripleGunBox`.<br>• Set `visbible` to true.<br>• Set `destroyed` to false.<br>• Set `this.x` to x.<br>• Set `y` to `-height -(random nuber from [0,500) ).`<br>• Set `collideDamage` to 0. |

#### 3.5.19.2 Method

| | |
|---|---|
| +void draw(GraphicsContext gc) | Draw TrippleGun box onto screen.<br>• Draw `RenderableHolder.tripleGunBox` at `x` and `y`. |
| +void onCollision(CollidableEntity others) | Update this object when collide with other.<br>• Set `hp` to 0.<br>• Set `destroyed` to `true`.<br>• Set `visible` to `false`. |

## 3.6 Package sharedObject

### 3.6.1 Interface IRenderable

#### 3.6.1.1 Method

| | |
|---|---|
| +int getZ(); | • Return this object z value. |
| +void draw(GraphicsContext gc) | • Draw this object onto Canvas. |
| +boolean isDestroyed() | • Return true if this object is detroyed.<br>• Ese return false. |
| +boolean isVisible() | • Return true if this object is visible.<br>• Else return false. |

### 3.6.2 Class RenderableHolder

#### 3.6.2.1 Field

| | |
|---|---|
| -RenderableHolder instance | A new instace for calling method in RenderableHolder. |
| -List<IRenderable> entities | A list that store all IRenderable object of the game. |
| -Comparator<IRenderable> comparator | A comparator that compare `IRenderable.` |

| | |
|---|---|
| +Image ship1 | A first image of a player ship. |
| +Image ship2 | A secondimage of a player ship. |
| +Image ship3 | A thirdimage of a player ship. |
| +Image ship4 | A fourthimage of a player ship. |
| +Image ship5 | A fifth image of a player ship. |
| +Image eBig | An image of a eBig enemy. |
| +Image eBoss | An image of a eBossenemy. |
| +Image eEyeball | An image of a eEyeball enemy. |
| +Image eBug | An image of a eBug enemy. |
| +Image eSquid | An image of a eSquid enemy. |
| +Image eWing | An image of a eWing enemy. |
| +Image bullet | An image of a laser bullet that a player will shoot. |
| +Image bossBullet | An image of a laser bullet that a eBoss will shoot. |
| +Image roundBulletB | An image of a laser bullet that a eEyeball will shoot. |
| +Image roundBulletY | An image of a laser bullet that a eWing will shoot. |
| +Image roundBulletR | An image of a laser bullet that a eBig will shoot. |
| +Image roundBulletP | An image of a laser bullet that a eSquid will shoot. |
| +Image missile | An image of a missile that player can shoot. |
| +Image exploArr[] | An array containing all explosion image that will be iterated to form a full explosive animation. |
| +Image background | A game's background image. |
| +Image hpBox | A hp bonus box image. |
| +Image tripleGunBox | A triple-gun bonus box image. |
| +Image missileBox | A missile bonus box image. |
| +AudioClip bgm | A background music. |
| +AudioClip laser | A laser shot sound effect. |
| +AudioClip explosion | An explosion sound effect. |
| +AudioClip explosion2 | Another explosion sound effect. |
| +AudioClip gameOverMusic | A music that will play when the game is over. |

| | |
|---|---|
| +AudioClip mainMenuMusic | A music that plays in main menu. |
| +AudioClip missileLaunch | A missile launching sound effect. |
| +Font inGameFont | A font used in a game text. |
| +Font inGameFontSmall | A smaller font use in a game text |
| +Image[] playership | An array of playership images. |
| +AudioClip[] explosions | An array of explosion sound effects. |

#### 3.6.2.2 Constructor

| | |
|---|---|
| +RenderableHolder() | Initialize entities and comparator<br>● Initilize `entites` as new synchronizedList ArrayList.<br>● Initialize `comparator` to compare between IRenderable 01, IRenderable 02.<br>○ if z value of 01 is more than z of 02 return 1.<br>○ else return -1. |

#### 3.6.2.3 Method

| | |
|---|---|
| +RenderableHolder getInstance() | ● return `instance`. |
| +void loadResource() | Load all images , fonts, and audio in /res folder<br>● Load all resources and store it in the field according to their name and type.<br>● Add all 12 explosion images to `exploArr[]`.<br>● Add ship1 - ship5 to `playership` array<br>● Add explosion and explosion2 to `explosions` array |
| +void add(IRenderable entity) | Add new Irenderable object ot entities and sort.<br>● Add entity to `entities`.<br>● Sort `entities` using `Collections.sort(entities, comparator)`. |
| +void update() | Update entities list<br>● Iterate IRenderable object through entities list<br>○ if the object is destroyed, then remove it. |
| +List<IRenderable> getEntities() | ● Return entities. |
| +void clear() | ● Clear entities. |

# 3.7 Package soundtrack

## 3.7.1 Class BackgroundMusic

### 3.7.1.1 Field

| | |
|---|---|
| -Thread bgmLoop | A thread use for looping background music. |
| -AudioClip bgm | The background music. |
| -final Task task; | A new task that is responsible for music looping method. |

### 3.7.1.2 Constructor

| | |
|---|---|
| +BackgroundMusic() | Create new bgmLoop thread to play background music.<br>● Initialized new Task to play music<br>In the task:<br>● Set `bgm` to `RenderableHolder.bgm`.<br>● Set `bgm` volume to 1.5.<br>● Set `bgm` cycleCount to `MediaPlayer.INDEFINITE`.<br>● Play `bgm`.<br>● Return null. |

### 3.7.1.3 Method

| | |
|---|---|
| +void startBackgroundMusic() | Start and play a background music indefinitely.<br>● Set `bgmLoop` to run a task.<br>● Start `bgmLoop`. |
| +void stopBackgroundMusic() | Stop background music.<br>● Stop `bgm`. |

# 3.8 Package window

## 3.8.1 Class SceneManager

### 3.8.1.1 Field

| | |
|---|---|
| -Stage primaryStage | The main Stage where every scene will be display on. |
| -Canvas mainMenuCanvas | The main menu Canvas .<br>● Initialized new `MainMenu`(). |
| -Scene mainMenuScene | The scene that will show MainMenu Canvas<br>● Initialized new Pane(mainMenuCanvas). |

| | |
|---|---|
| +SCENE_WIDTH | The height of the program windows.<br>● It is 600 pixel. |
| +SCENE_HEIGHT | The height of the program windows.<br>● It is 800 pixel. |

#### 3.8.1.2 Method

| | |
|---|---|
| +void initialize(Stage stage) | Initialized new stage and display it.<br>● Set `primaryStage` to stage.<br>● Show the `primaryStage`. |
| +void gotoMainMenu() | Initialized new main menu and display it.<br>● Set `primaryStage` to `mainMenuScene`.<br>● RequestFocus for `mainMenuCanvas`. |
| +void gotoSceneOf(Canvas canvas) | Change a scene to canvas scene and display it.<br>● Set primaryStage to<br>○ `new Scene(new Pane(canvas), 600, 800).`<br>● RequestFocus for `canvas`. |

## Source of images and audio.

- Main menu music : Main Theme (Star Fox 64) - Super Smash Bros. Brawl
- Background music : Corneria - Super Smash Bros. Brawl
- Gameover music : Final Fantasy I - Game Over
- Image sprites : https://www.deviantart.com

All media are used for acedamic purposes only as according to Fair Use policy.