Star Destroyer Documentation

Created by

Poravee Binhayeearason 6230314421

Supanart Barnsongkit 6230522621

2110215 Programming Methodology

Semester 1 Year 2020

Chulalongkorn University

Star_Destroyer

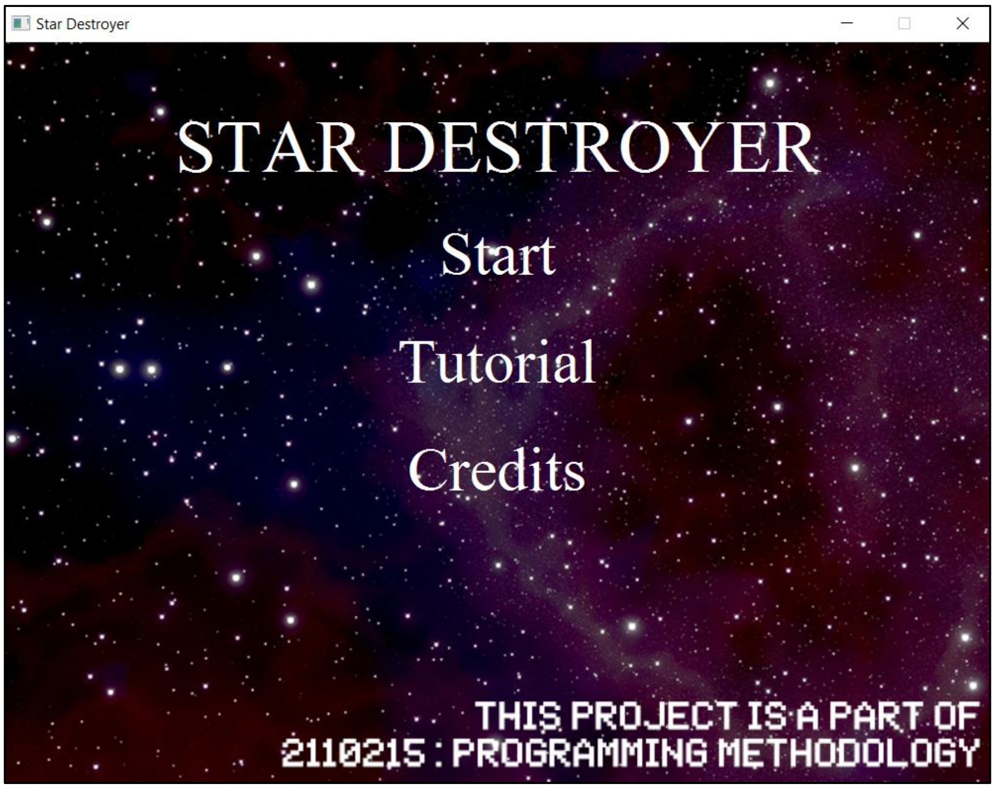
Introduction

Star_Destroyer is inspired by arcade game name "Space Invader". The objective of the game is to have the most score points.

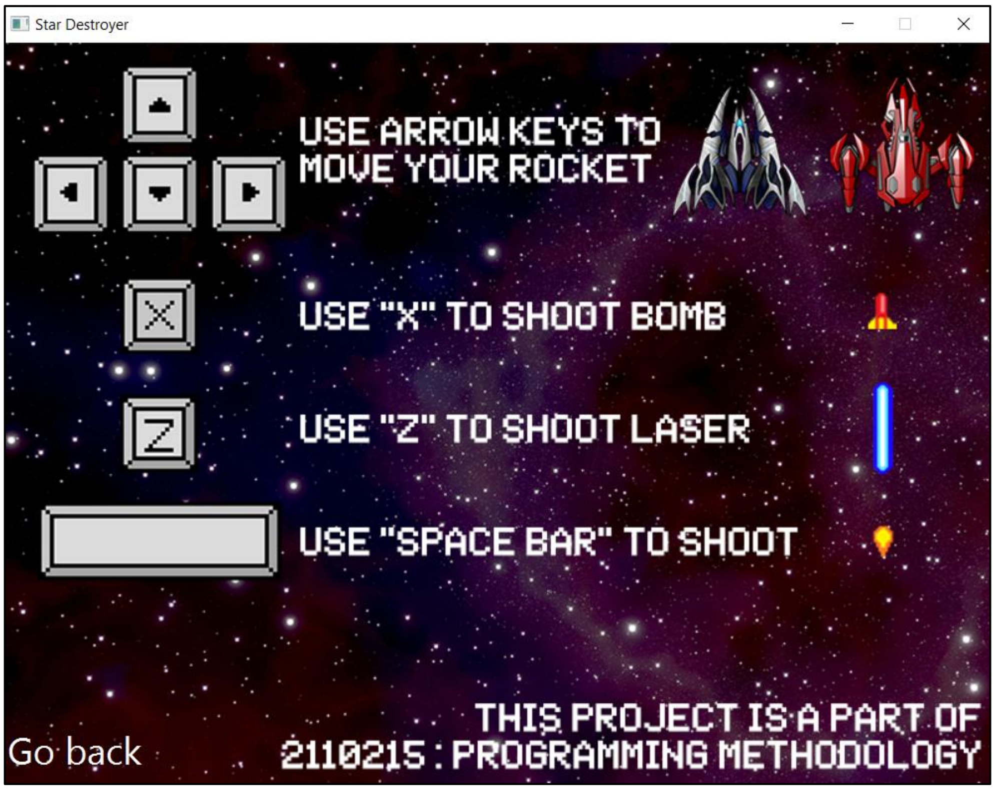
How to play

- Use arrow keys to move your rocket.
- Use key 'X' to shoot bomb.
- Use key 'Z' to shoot laser.
- Use key 'SPACE' to shoot normal bullet.
- Defeat invaders and survive to protect our galaxy.

Title scene



Tutorial scene



USE ARROW KEYS TO MOVE YOUR ROCKET

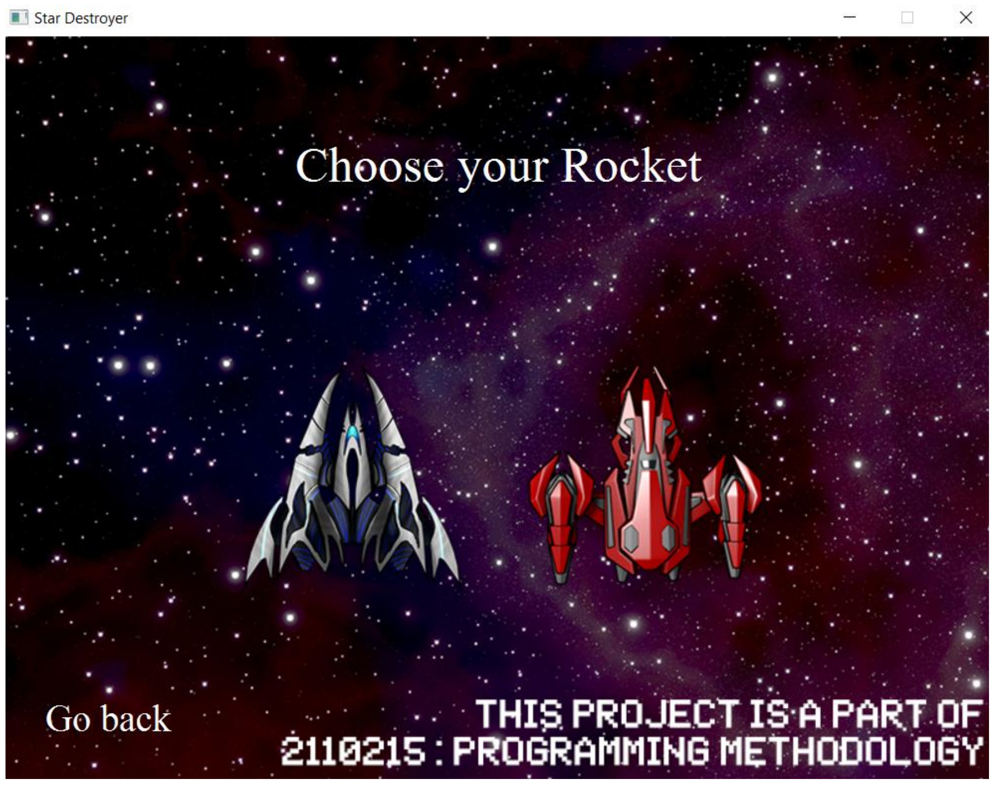
USE "X" TO SHOOT BOMB

USE "Z" TO SHOOT LASER

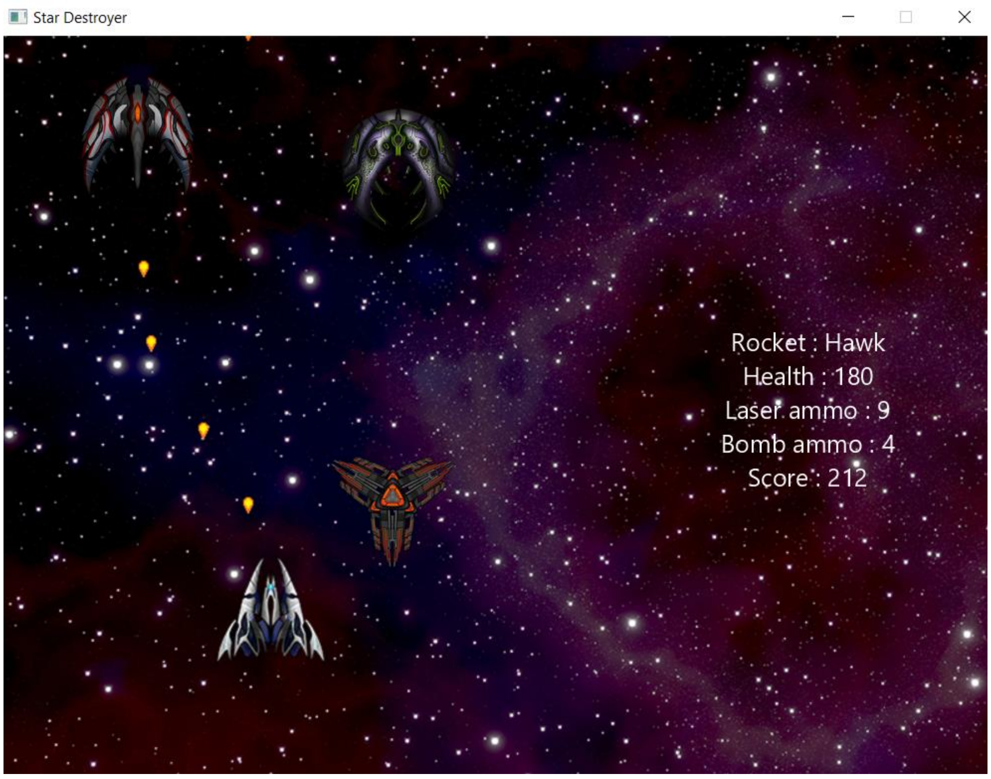
USE "SPACE BAR" TO SHOOT

THIS PROJECT IS A PART OF 2110215 : PROGRAMMING METHODOLOGY

ChooseRocket scene

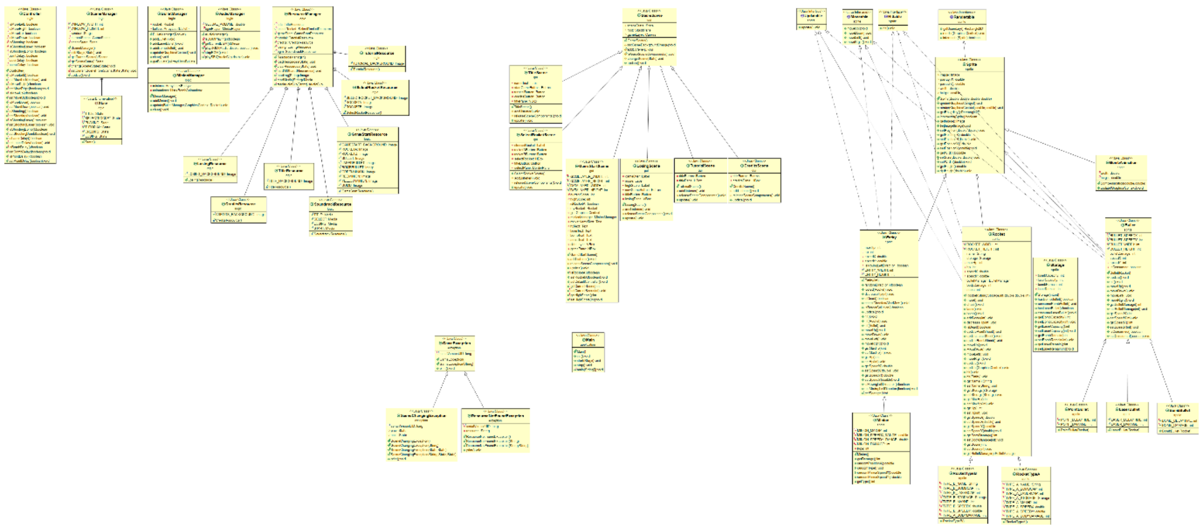


GameStart scene



Rocket : Hawk
Health : 180
Laser ammo : 9
Bomb ammo : 4
Score : 212

Class diagram



1.Package sprite

1.1 interface Moveable

1.1.1 Methods

+ void moveUp();	Move the sprite up
+ void moveDown();	Move the sprite down
+ void moveLeft();	Move the sprite left
+ void moveRight();	Move the sprite right

1.2 interface Hittable

1.1.2 Methods

+ void hit();	To do something depending on what the object is

1.3 interface Updatable

1.1.3 Methods

+ void update();	To update the sprite

1.4 interface Renderable

1.1.4 Methods

+ Rectangle2D getBoundary();	To get object boundary
+ void render(GraphicsContext gc);	To draw object in scene
+ Boolean intersects(Sprite s);	To check the sprite is intersected with another sprite

1.5 class Sprite implements Renderable

1.5.1 Fields

# Image image;	Image of the sprite
# double positionX;	Position (X-axis) of the sprite
# double positionY;	Position (Y-axis) of the sprite
# double width;	Sprite width
# double height;	Sprite height

1.5.2 Constructors

+ void Sprite(double positionX, double positionY, double width, double height	Set position and size

1.5.3 Methods

+ void render(GraphicsContext gc);	Draw image
+ void render(GraphicsContext gc, double width, double height);	Draw image and fix size
+ Rectangle2D getBoundary();	return object boundary
+ Boolean intersects(Sprite s);	To check the sprite is intersected with another sprite
Getters & Setters	To get and set field

1.6 class Bullet extends Sprite implements Moveable, Hittable, Updatable

1.6.1 Fields

+ static final int BULLET_SPEEDX;	BULLET_SPEEDX = 0
+ static final int BULLET_SPEEDY;	BULLET_SPEEDX = 10
+ static final int BULLET_WIDTH;	BULLET_WIDTH = 10
+ static final int BULLET_HEIGHT;	BULLET_HEIGHT = 10
- int bulletDamage;	Damage of the bullet
- int speedX;	Speed (X-axis) of bullet
- int speedY;	Speed (Y-axis) of bullet
- Boolean isConsumed;	To consume (disappear) when hit minion (expects LaserBullet)

1.6.2 Constructor

+ Bullet(Rocket rocket);	- Construct super class with default values (0) - set speed - set isConsumed to false

1.6.3 Methods

+ void update();	moveUp();
+ void hit();	Set isConsumed to true
+ void moveUp();	position -= speedY;
+ void moveDown();	Do nothing
+ void moveLeft();	Do nothing
+ void moveRight();	Do nothing
Getters & Setters	To get and set field

1.7 class PointBullet extends Bullet

1.7.1 Fields

+ static final int POINT_DELAYTIME;	POINT_DELAYTIME = 50
+ static final int POINT_DAMAGE;	POINT_DAMAGE = 5

1.7.2 Constructors

+ PointBullet(Rocket rocket);	- Construct super class with rocket - set damage - set image - set size - set position at rocket position

1.8 class LaserBullet extends Bullet

1.8.1 Fields

+ static final int LASER_DELAYTIME;	LASER_DELAYTIME = 3000
+ static final int LASER_DAMAGE;	LASER_DAMAGE = 40

1.8.2 Constructors

+ LaserBullet(Rocket rocket);	- Construct super class with rocket - set damage - set image - set size - set position at rocket position

1.9 class BombBullet extends Bullet

1.9.1 Fields

+ static final int BOMB_DELAYTIME;	BOMB_DELAYTIME = 3000
+ static final int BOMB_DAMAGE;	BOMB_DAMAGE = 50

1.9.2 Constructors

+ BombBullet(Rocket rocket);	- Construct super class with rocket - set damage - set image - set size - set position at rocket position

1.10 class BombAnimation extends Sprite implements updatable

1.10.1 Fields

- static double width;	width = 50
- static double height;	height = 50

1.10.2 Constructors

+ BombAnimation(double positionX, double positionY);	- Construct super class with positionX, positionY, width, height - Set size to (50,50)

1.10.3 Methods

+ void update();	Do nothing
+ void update(GraphicContext gc);	Render and play SFX

1.11 class Storage

1.11.1 Fields

- int bombCapacity	The amount of Bomb Bullets when start game
- int laserCapacity	The amount of Laser Bullets when start game
- int bombRemain	Current quantity of Bomb Bullets
- int laserRemain	Current quantity of Laser Bullets

1.11.2 Constructors

+ Storage(int bombCapacity, int laserCapacity);	- set bombCapacity, laserCapacity - set bombRemain, laserRemain (equals their capacities)

1.11.3 Methods

+ boolean hasBombBullet();	To check bombRemain > 0
+ void consumeBombBullet();	Decrease bombRemain by 1
+ boolean hasLaserBullet();	To check laserRemain > 0
+ void consumeLaserBullet();	Decrease laserRemain by 1
Getters & Setters	To get and set field

1.12 class Rocket extends Sprite implements Moveable, Hittable, Updatable

1.12.1 Fields

+ static final int ROCKET_WIDTH;	ROCKET_WIDTH = 100
+ static final int ROCKET_HEIGHT;	ROCKET_HEIGHT = 100
- String name;	Name of the rocket
- Storage storage;	Rocket storage (contains bullets)
- int maxHp;	Max Hp of the rocket
- int hp;	Current Hp of the rocket
- double speedX;	Rocket speed (X-axis) when move
- double speedY;	Rocket speed (Y-axis) when move
- BulletManager bulletManager;	Bullet manager of the rocket
- int bodyDamage;	Damage the entity get when hit the rocket
- int score;	Rocket score

1.12.2 Constructors

+ Rocket(String name, Storage storage, int maxHp, double speedX, double speedY, int bodyDamage);	- Construct super class with default values (0) - set name and bodyDamage - set maxHp, hp by maxHp value - set speed (X and Y) - set score to 0 - set bulletManager (use bulletManager constructor)

1.12.3 Methods

+ void shoot();	Add pointBullet in bulletManager
+ void laser();	Add laserBullet in buleltManager
+ void bomb();	Add bombBullet in bulletManager
+ void addScore(int score);	Increase current score
+ void decreaseHp(int damage);	Decrease current Hp by damage (current Hp is more or equal 0)
+ Boolean isDead();	To check current Hp is 0
+ void moveUp();	positionY -= speedY;
+ void moveDown();	positionY += speedY;
+ void moveLeft();	positionX -= speedX;
+ void moveRight();	positionX += speedX;
+ void move();	Check with Controller class and GameStartScene.GAMELAYER_HEIGHT and WIDTH to move (use interface Moveable methods)
+ void updatePointShoot();	If Controller.isShooting() and not isPointDelay(); then start thread to - shoot(); - play sound effect “gunsound.wav” - set pointDelay to true - thread sleep(POINT_DELAYTIME) - set pointDelay to false
+ void updateLaserShoot();	If Controller.isShootingLaser() and not isLaserDelay(); then start thread to - laser(); - play sound effect “lasersound.wav” - set laserDelay to true - thread sleep(LASER_DELAYTIME) - set laserDelay to false

+ void updateBombShoot();	If Controller.isShootingBomb() and not isBombrDelay(); then start thread to - bomb(); - play sound effect “bombsound.wav” - set bombDelay to true - thread sleep(BOMB_DELAYTIME) - set bombDelay to false
+ void update();	Do nothing
+ void update(GraphicsContext gc);	- move(); - updatePointShoot(); - updateLaserShoot(); - updateBombShoot(); - bulletManager.update(gc); - render(gc);
+ void hit();	Do nothing
+ void hit(Entity entity);	Decrease Hp by entity’s damage
Getters & Setters	To get and set field

1.13 class RocketTypeA extends Rocket

1.13.1 Fields

- static final String TYPE_A_NAME;	TYPE_A_NAME = “Hawk”
- static final int TYPE_A_BOMBCAP;	TYPE_A_BOMBCAP = 5
- static final int TYPE_A_LASERCAP;	TYPE_A_LASERCAP = 10
- static final Storage TYPE_A_STORAGE;	TYPE_A_STORAGE = new Storage(TYPE_A_BOMBCAP, TYPE_A_LASERCAP);
- static final int TYPE_A_MAXHP;	TYPE_A_MAXHP = 250

- static final double TYPE_A_SPEEDX;	TYPE_A_SPEEDX = 6
- static final double TYPE_A_SPEEDY;	TYPE_A _SPEEDY = 6
- static final int TYPE_A_BODYDAMAGE;	TYPE_A_BODYDAMAGE = 20

1.13.2 Constructors

+ RocketTypeA();	- Construct super class with its constants - set image - set width and height - set positionX to GAMELAYER_WIDTH/2 – this.getWidth() - set positionY to GAMELAYER_HEIGHT – this.getHeight()

1.14 class RocketTypeB extends Rocket

1.14.1 Fields

- static final String TYPE_B_NAME;	TYPE_B_NAME = “Tank”
- static final int TYPE_B_BOMBCAP;	TYPE_B_BOMBCAP = 10
- static final int TYPE_B_LASERCAP;	TYPE_B_LASERCAP = 15
- static final Storage TYPE_B_STORAGE;	TYPE_B_STORAGE = new Storage(TYPE_B_BOMBCAP, TYPE_B_LASERCAP);
- static final int TYPE_B_MAXHP;	TYPE_B_MAXHP = 350

- static final double TYPE_B_SPEEDX;	TYPE_B_SPEEDX = 4
- static final double TYPE_B_SPEEDY;	TYPE_B _SPEEDY = 4
- static final int TYPE_B_BODYDAMAGE;	TYPE_B_BODYDAMAGE = 30

1.14.2 Constructors

+ RocketTypeB();	- Construct super class with its constants - set image - set width and height - set positionX to GAMELAYER_WIDTH/2 – this.getWidth() - set positionY to GAMELAYER_HEIGHT – this.getHeight()

1.15 class Entity extends Sprite implements Moveable, Hittable, Updatable

1.15.1 Fields

+ static final int ENTITY_WIDTH;	ENTITY_WIDTH = 100
+ static final int ENTITY_HEIGHT;	ENTITY_HEIGHT = 100
- int maxHp;	Max Hp of the entity
- int hp;	Current Hp of the entity
- double speedX;	Speed (X-axis) of the entity
- double speedY;	Speed (Y-axis) of the entity
- Boolean isMovingLeftDirection;	To check direction of the entity (X-axis)

1.15.2 Constructors

+ Entity(int maxHp);	- Construct super class with default values (0) - set maxHp - set current hp - set moveLeftDirection (use randomDirection())

1.15.3 Methods

+ Boolean randomDirection();	Return new Random().nextBoolean();
+ void looted(Rocket rocket);	Add rocket score (score is random int from 1 to 10)
+ void decreaseHp(int damage);	Decrease current Hp by damage (current Hp is more or equal 0)
+ Boolean isDead();	To check current hp equals 0
+ isBorderCollision();	To check positionX is between 0 and GAMELAYER_WIDTH – this.getWidth()
+ void moveUp();	Do nothing
+ void moveDown();	positionY += speedY
+ void moveLeft();	positionX -= speedX
+ void moveRight();	positionX += speedX
+ void checkDirectionAfterMove();	If isBorderCollision then switch isMovingLeftDirection (true to false, false to true)
+ void update();	- moveDown(); - if isMovingLeftDirection then moveLeft() ,otherwise moveRight() - checkDirectionAfterMove();
+ void hit();	Do nothing

+ void hit(Rocket rocket);	Decrease Hp by rocket’s bodyDamage
+ void hit(Bullet bullet);	Decrease Hp by bullet’s damage
+ int getDamage();	To be overrided in Minion class
Getters & Setters	To get and set field

1.16 class Minion extends Entity

1.16.1 Fields

- static final int MINION_MAXHP;	MINION_MAXHP = 20
- static final double MINION_SPEEDX_RANGE;	MINION_SPEEDX_RANGE = 3;
- static final double MINION_SPEEDY_RANGE;	MINION_SPEEDY_RANGE = 3;
- static final int MINION_DAMAGE;	MINION_DAMAGE = 10;
- int type;	Type of Minion (used to indicate minion’s color)

1.16.2 Constructors

+ Minion();	- Construct super class with its constants - randomType(); - set speedX (use randomMinionSpeedX()) - set speedY (use randomMinionSpeedY()) - if type is 1 then set image to green minion else if type is 2 then set image to yellow minion else if type is 3 then set image to red minion

	- set positionX (use randomPositionX())

1.16.3 Methods

+ int getDamage();	Return MINION_DAMAGE (Override method from Entity class)
+ double randomPositionX();	Return random double from 0 to GAMELAYER_WIDTH – getWidth()
+ void randomType();	Set type to random int from 1 to 3
+ double randomMinionSpeedX();	Return new Random().nextDouble() * MINION_SPEEDX_RANGE + 1;
+ double randomMinionSpeedY();	Return new Random().nextDouble() * MINION_SPEEDY_RANGE + 1;
+ int getType();	Return type

2.Package logic

2.1 class Controller

2.1.1 Fields

- static boolean isMoveLeft;	isMoveLeft = false
- static boolean isMoveRight;	isMoveRight = false
- static boolean isMoveUp;	isMoveUp = false
- static boolean isMoveDown;	isMoveDown = false
- static boolean isShooting;	isShooting = false
- static boolean isShootingLaser;	isShootingLaser = false
- static boolean isShootingBomb;	isShootingBomb = false
- static boolean pointDelay;	pointDelay = false
- static boolean laserDelay;	laserDelay = false
- static boolean bombDelay;	bombDelay = false

2.1.2 Methods

Getters & Setters	To get and set fields

2.2 class BulletManager

2.2.1 Fields

- Rocket rocket;	Choosed rocket in game
- ArrayList<Bullet> bullets	Bullets = new ArrayList<>();

2.2.2 Constructors

+ BulletManager(Rocket rocket);	Set rocket

2.2.3 Methods

+ void addBullet();	Add new pointBullet in bullets
+ void addLaserBullet();	Add new lasetBullet in bullets
+ void addBombBullet();	Add new bombBullet in bullets
+ void update(GraphicsContext gc);	- Initailize ArrayList<Integer> toRemove - check all bullet in bullets if its position less than 0 then add bullet index to toRemove, otherwise update and render it - sort toRemove (reverseOrder) - remove bullet in bullets by its index for all bullet in toRemove
+ void clear();	Remove all bullet in bullets (use clear())
Getters & Setters	To get and set fields

2.3 class MinionManager

2.3.1 Fields

- ArrayList<Entity> minions	Minions = new ArrayList<>();
- List<BombAnimation> animations	Animations = Collections.synchronizeList(new ArrayList<BombAnimation>());

2.3.2 Constructors

+ MinionManager();	Calls addMinion() 3 times

2.3.3 Methods

+ void addMinion();	Add minion in minions
+ void update(BulletManager bulletmanager, GraphicsContext gc, Rocket rocket);	- Initailize ArrayList<Interger> toRemoveBullets and toRemoveMinions - int more = 0 - for all minion in minions + update and render + if position is out of scene then add to toRemoveMinions + if minion intersects rocket then call hit both sprites + check with all bullet in bullets if minion intersects bullet then call hit both sprites (except laserBullet), if bullet is BombBullet then initialize bombanimation and add to animations and start thread which sleep 50 millisecond then remove animation and check if bullet is Consumed then add to toRemoveBullets - if minion is dead then add to toRemoveMinions - update all bombanimation in animations - Initailize HashSet<Integer> to remove each minion in minions and bullet in bullets - when minion removed, add more with 1 - addMinion according to more times
+ void clear	Clear minions

2.4 class AudioManager

2.4.1 Fields

- static final double GLOBAL_VOLUME;	GLOBAL_VOLUME = 0.5
- static MediaPlayer bgmPlayer	The object to play BGM

2.4.2 Methods

- static boolean isBGMPlaying();	To check bgmPlayer is not null
+ static Media getCurrentBGM();	To get current BGM
+ static void playBGM(Media bgm, double localVolume, Boolean isLoop);	- if bgm is null then stopBGM - if bgm is not equals current BGM then play bgm instead
+ static void stopBGM();	- if BGM is playing then stop and set to null

2.5 class ResourceManager

2.5.1 Fields

+ static TitleResource title;	The innerclass that contains resource for title scene
+ static SelectRocketResource selectRocket;	The innerclass that contains resource for selectRocket scene
+ static GameStartResource gameStart;	The innerclass that contains resource for gameStart scene
+ static TutorialResource tutorial;	The innerclass that contains resource for tutorial scene
+ static CreditsResource credits;	The innerclass that contains resource for credits scene
+ static LosingResource losing;	The innerclass that contains resource for losing scene
+ static SoundtrackResource bgm;	The innerclass that contains sound for all scenes

2.5.1.1 innerclass TitleResource

2.5.1.1.1 Fields

+ final Image TITLE_BACKGROUND;	Background of the title scene

2.5.1.1.2 Constructors

+ TitleResource();	Set TITLE_BACKGROUND

2.5.1.2 innerclass SelectRocketResource

2.5.1.2.1 Fields

+ final Image SELECTROCKET_BACKGROUND;	Background of the select rocket scene
+ final Image ROCKETA;	Rocket type A image
+ final Image ROCKETB;	Rocket type B image

2.5.1.2.2 Constructors

+ SelectRocketResources();	Set Image in all fields

2.5.1.3 innerclass GameStartResource

2.5.1.2.1 Fields

+ final Image GAMESTART_BACKGROUND;	Background of the gamestart scene
+ final Image ROCKETA;	Rocket type A image
+ final Image ROCKETB;	Rocket type B image
+ final Image BULLET	Point bullet image
+ final Image LASERBULLET	Laser bullet image
+ final Image BOMBBULLET	Bomb bullet image
+ final Image GREENMINION	Green minion image
+ final Image REDMINION	Red minion image
+ final Image YELLOWMINION	Yellow image
+ final Image BOMB	Bomb effect image

2.5.1.2.2 Constructors

+ GameStartResource();	Set Image in all fields

2.5.1.4 innerclass TutorialResource

2.5.1.4.1 Fields

+ final Image TUTORIAL_BACKGROUND	Background of the tutorial scene

2.5.1.4.2 Constructors

+ TutorialResource();	Set Image in all fields

2.5.1.5 innerclass CreditsResource

2.5.1.4.1 Fields

+ final Image CREDITS_BACKGROUND	Background of the credits scene

2.5.1.4.2 Constructors

+ CreditsResource();	Set Image in all fields

2.5.1.6 innerclass LosingResource

2.5.1.4.1 Fields

+ final Image LOSING_BACKGROUND	Background of the losing scene

2.5.1.4.2 Constructors

+ LosingResource();	Set Image in all fields

2.5.1.7 innerclass SoundtrackResource

2.5.1.7.1 Fields

+ final Media TITLE;	BGM in title scene
+ final Media SELECT	BGM in select rocket scene
+ final Media START	BGM in game start scene
+ final Media LOSING	BGM in losing scene

2.5.1.7.2 Constructors

# Soundtrack Resource	- read media in all fields

2.5.2 Methods

+ static void loadResources(SceneManager.State sceneState);	Initailize 1 in 6 subclasses resources followed by sceneState (if catch exception then throws new ResourceNotFoundException (e.getMessage())
+ static void clearResources(SceneManager.State sceneState);	Set 1 in 6 subclasses resources to null followed by sceneState
+ static void loadAllSharedResources();	If bgm is null then set to new SoundtrackResource();
+ static Image readImg(String filename);	return new Image(ClassLoader.getSystemResource(filename).toString());
+ static Media readMedia(String filename);	return new Image(ClassLoader.getSystemResource(filename).toString());
+ static AudioClip readAudioClip(String filename);	return new AudioClip(ClassLoader.getSystemResource(filename).toString()) ;

2.6 class SceneManager

2.6.1 Fields

+ static final int WINDOW_WIDTH ;	WINDOW_WIDTH = 800
+ static final int WINDOW_HEIGHT ;	WINDOW_HEIGHT = 600
- static Stage window;	Game Stage
- static GameScene currentScene;	Game current scene
- static State sceneState;	Game current state
+ static enum State;	{TITLE, SELECTROCKET, PLAYING, TUTORIAL, CREDITS, LOSING}

2.6.2 Methods

+ static void init(Stage stage, State sceneState) throws GameException;	- set window to stage - load all shared resources - set sceneStage - set current scene from sceneStage - set scene in window to currentScene
+ static Scene getCurrentScene();	Return currentScene
+ static State getSceneState;	Return sceneState
+ static void changesSceneState(State sceneState) throws GameException;	- if window is null then throw new SceneChangingException ("At SceneManager, window is null"); - otherwise, set sceneState, set scene on window and show

+ static void setCurrentSceneFromSceneState (State sceneState) throws GameException;	- set currentScene to null - load resources and set currentScene to 1 in 6 scene followed by sceneState
+ static void update();	Update currentScene if it is not null

3.Package exception

3.1 class GameException

3.1.1 Fields

- static final long serialVersionUID;	serialVersionUID = -2048416760537782547L

3.1.2 Constuctors

+ GameException();	Construct super class
+ GameException(String message);	Construct super class with message

3.1.3 Methods

+ void print();	To print exception

3.2 class ResourceNotFoundException extends GameException

3.2.1 Fields

- static final long serialVersionUID;	serialVersionUID = 5261600905049703426L
- static String resource;	The resource that not found

3.2.2 Constuctors

+ ResourceNotFoundException ();	Construct super class
+ ResourceNotFoundException (String resource);	Construct super class and set resource
+ ResourceNotFoundException (String resource, String message);	Construct super class with message and set resource

3.2.3 Methods

+ void print()	To print exception

3.3 class SceneChangingException extends GameException

3.3.1 Fields

- static final long serialVersionUID;	serialVersionUID = -9107197163021617917L
# SceneManager.State prev;	Previous stage
# SceneManager.State next;	Next stage

3.3.2 Constuctors

+ SceneChangingException ()	Construct super class and set prev and next to null
+ SceneChangingException (String message)	Construct super class with message and set prev and next to null
+ SceneChangingException (SceneManager.State prev, SceneManager.State next)	Construct super class and set prev and next

+ SceneChangingException(String message, SceneManager.State prev, SceneManager.State next)	Construct super class with message and set prev and next

3.3.3 Methods

+ void print()	To print exception

4.Package gui

4.1 class GameScene extends Scene

4.1.1 Fields

# SceneManager.State sceneState;	Scene state
# StackPane root;	Root of the scene
# Canvas gameLayer	Canvas of the scene

4.1.2 Constructors

+ GameScene();	- Construct super class with new StackPane - set sceneState to null - initialize gameLayer to new Canvas by WINDOW_WIDTH and WINDOW_HEIGHT - set root to this.getRoot() - set root prefer size to window width and height

4.1.3 Methods

# void addListener();	To add event handling in scene
# void releaseSceneComponents();	To clear resources in scene
# void changScene(SceneManager.State sceneState);	- releases scene components - try change scene state, if catch exception then print exception and try set scene to current scene, if catch exception again then print that exception
+ void update();	Do nothing

4.2 class TitleScene extends GameScene

4.2.1 Fields

- Label nameText;	The lable which shows game name
- Button startGameButton;	Game start button (to go SelectRocketScene)
- Button tutorialButton;	Tutorial button (to go tutorial)
- Button creditsButton;	Credits button (to go credits)
- VBox titlePane;	Pane which contains components

4.2.2 Constructors

+ TitleScene();	- Constructs super class - Set sceneState to State.TITLE - Set game background - Initialize nameText - Initialize startGameButton - Initialize tutorialButton - Initialize creditsButton - add listener - Initialize titlePane and add label and buttons in it - Add titlePane in root - Play BGM

4.2.3 Methods

# void addListener();	Set event handling to all buttons (use changeScene(State to change))
# void releaseSceneComponents();	Clear resources in scene
+ void update();	Do nothing

4.3 class SelectRocketScene extends GameScene

4.3.1 Fields

- Label chooseRocket;	The lable which tells player to choose rocket
- Button rocketAButton;	Button to choose rocket type A
- Button rocketBButton;	Button to choose rocket type B
- HBox selectRocket	Contains two rocketButtons
- Button titleButton	Title button (to go title)
- BorderPane selectPane;	Pane which contains components

4.3.2 Constructors

+ SelectRocketScene();	- Constructs super class - Set sceneState to State.SELECTROCKET - Set game background - Initialize chooseRocket - Initialize rocketAButton - Initialize rocketBButton - Initialize selectRocket and add two buttons in it - Initialize titleButton - add listener - Initialize selectPane and add all components in it - Add selectPane in root - Play BGM

4.3.3 Methods

# void addListener();	Set event handling to all buttons (use changeScene(State to change) and GameStartScene.setRocket(A/B) to true)
# void releaseSceneComponents();	Clear resources in scene
+ void update();	Do nothing

4.4 class GameStartScene extends GameScene

4.4.1 Fields

+ static final int GAMELAYER_WIDTH;	GAMELAYER_WIDTH = 500
+ static final int GAMELAYER_HEIGHT;	GAMELAYER_HEIGHT = 600
+ static final int DATALAYER_WIDTH;	DATALAYER_WIDTH = 300
+ static final int DATALAYER_HEIGHT;	DATALAYER_HEIGHT = 600
- static int currentScore;	Current score in game
- static int highScore	highScore = 0
- static Boolean isRocketA	isRocketA = false (true when choose rocket type B)
- Rocket myRocket;	Rocket in game
- GraphicsContext gc;	Graphic in game
- MinionManager minionManager;	To manage minions in game
- Text rocketNameText;	Shows rocket name
- Text hpText;	Shows rocket current Hp
- Text laserText;	Shows remain laser bullets
- Text bombText;	Shows remain bomb bulelts
- Text scoreText;	Shows current score
- VBox dataLayer;	Contains all texts
- HBox gamePane;	Contains gameLayer and dataLayer

4.4.2 Constructors

+ GameStartSccene();	- Constructs super class - Set sceneState to State.PLAYING - Set game background - Initialize gameLayer - set gc to gameLayer.getGraphicsContext2D(); - if isRocketA set myRocket to RocketTypeA otherwise, set to RocketTypeB then render it - Initialize all texts - Initialize dataLayer and add all texts in it - add listener - Initialize gamePane and add all gameLayer and dataLayer in it - Add gamePane in root - Initialize minionManager - set default controller - set current score to 0 - Play BGM

4.4.3 Methods

# void addListener();	Set event handling to scene
# void releaseSceneComponents();	Clear resources in scene
+ void update();	- If rocket is dead then change scene to losing scene - Otherwise, update all components and render
+ void setDefaultController();	Set all fields in controller to false
Getters & Setters	To get and set fields

4.5 class LosingScene extends GameScene

4.5.1 Fields

- Label gameOver;	To show text game over
- Label score;	To show your score
- Lable highScore;	To show your highscore
- Button newGameButton;	To play again
- Button titleButton;	To go title scene
- VBox losingPane	Pane which contains components

4.5.2 Constructors

+ LosingScene();	- Constructs super class - Set sceneState to State.LOSING - Set game background - Initialize gameOver, score, highscore - Initialize titleButton and newGameButton - Initailize losingPane and add all labels and buttons in it - add listener - Add losingPane in root - Play BGM

4.5.3 Methods

# void addListener();	Set event handling to all buttons
# void releaseSceneComponents();	Clear resources in scene
+ void update();	Do nothing

4.6 class TutorialScene extends GameScene

4.6.1 Fields

- Button titleButton;	To go Title scene
- HBox titlePane;	Contains titleButton

4.6.2 Constructors

+ TutorialScene();	- Constructs super class - Set sceneState to State.TUTORIAL - Set game background with tutorial background - Initialize titleButton - Initailize titlePane and add titleButton in it - add listener - Add titlePane in root - Play BGM

4.6.3 Methods

# void addListener();	Set event handling to all buttons
# void releaseSceneComponents();	Clear resources in scene
+ void update();	Do nothing

4.7 class CreditsScene extends GameScene

4.7.1 Fields

- Button titleButton;	To go Title scene
- HBox creditsPane;	Contains titleButton

4.6.2 Constructors

+ TutorialScene();	- Constructs super class - Set sceneState to State.CREDITS - Set game background with credits background - Initialize titleButton - Initailize titlePane and add titleButton in it - add listener - Add titlePane in root - Play BGM

4.6.3 Methods

# void addListener();	Set event handling to all buttons
# void releaseSceneComponents();	Clear resources in scene
+ void update();	Do nothing

5.Package application

5.1 class Main extends Application

5.1.2 Methods

+ void init();	Do nothing
+ void start(Stage primaryStage) throws Exception;	- Initialize scene manager with primaryStage and State.TITLE - set primaryStage title - set primaryStage resizable to false - show primaryStage - Initialize AnimationTimer contains SceneManager .update() in handle method then start AnimationTimer
+ void stop();	Do nothing
+ static void main(String[] args);	Launch application