Homework 1

Report

Murat Yıldız

1801042004

Drawable

To something to be drawable, it has to implement draw method

Updatable

To something to be updatable, it has to implement update method

Collisinable

To something to be collisinable, it has to implement setCollisionBox and getCollisionBox method

GameObject

To something to be a GameObject, it has to implement all Drawable, Updatable, Collisinable interfaces

Main character

Main character has a jumpBehaviour so can be changed at run time from jumpHigh to jumpLow or vice versa.

Main character is a GameObject.

Main character has a life counter which initially 3.

Main character has a power up.

Main character has a collision box which is rectangle to understand whether it collides with other game objects.

Main character has an image that will be draw.

Obstacle

Obstacle is a GameObject.

Obstacle has a collision box which is rectangle to understand whether it collides with other game objects.

Obstacle has a point box which is rectangle to understand whether character jumped over obstacle or not

Land

Land has images to create a land.

Land has landscapes which an image and x cordinate of that image.

PowerUp

To something to be a PowerUp, it has to implemenet getMultiplier method.

PowerUpDecorator

PowerUpDecorator inherits PowerUp and getDescription method abstract.

Any class inherit from PowerUpDecorator must implement getMultiplier method and getDescription method

PowerUpA

PoweUpA is a GameObject and it inherits PowerUpDecorator so it is a PowerUp.

PoweUpA has a PowerUp.

PowerUpB

PoweUpB is a GameObject and it inherits PowerUpDecorator so it is a PowerUp.

PoweUpB has a PowerUp.

PowerUpC

PoweUpC is a GameObject and it inherits PowerUpDecorator so it is a PowerUp.

PoweUpC has a PowerUp.

PowerUpD

PoweUpD is a GameObject and it inherits PowerUpDecorator so it is a PowerUp.

PoweUpD has a changeJumpMode function that returns the changed jump mod.

Game Design

Key Actions

There is a key listener in game screen.
When a Space bar key pressed it triggers the keyPressed method, we check whether character is on ground before do action jump. If it is on ground , character calls its performJump.

When a D key is pressed it triggers the keyPressed method, we change moveRight value to true until D key released. After D key released it triggers the keyReleased method, we change moveRight value to false.

Update Actions

If character is alive and move to right we count it as a frame and every frame we do;

We call all game objects update method which will update x cordinates of collisionBox in gameObject list.

Remove if any game object which is out of screen from gameObject list.

Randomly between 10-30 frame a game object (obstacle or powerup) will generated and added to gameObjects list.

Call character update method which update x cordinate of collisionBox and applied gravity (increment y cordinate at constant speed).

We look every game object in gameObjects list whether it collides with character.

If a powerup collides, it will disappear and removed, added to character's powerup field in Decorator pattern way.

If an obstacle collides, it will disappear and removed, character's life will decremented.

We look every obstacle's pointBox in gameObject list to know character gain point or not.
If character gain points, it multiplied by character's powerup's getMultiplier() method

Draw Actions

Call land's draw method.

Call all gameObjects draw method.

Call character's draw method

Notes:

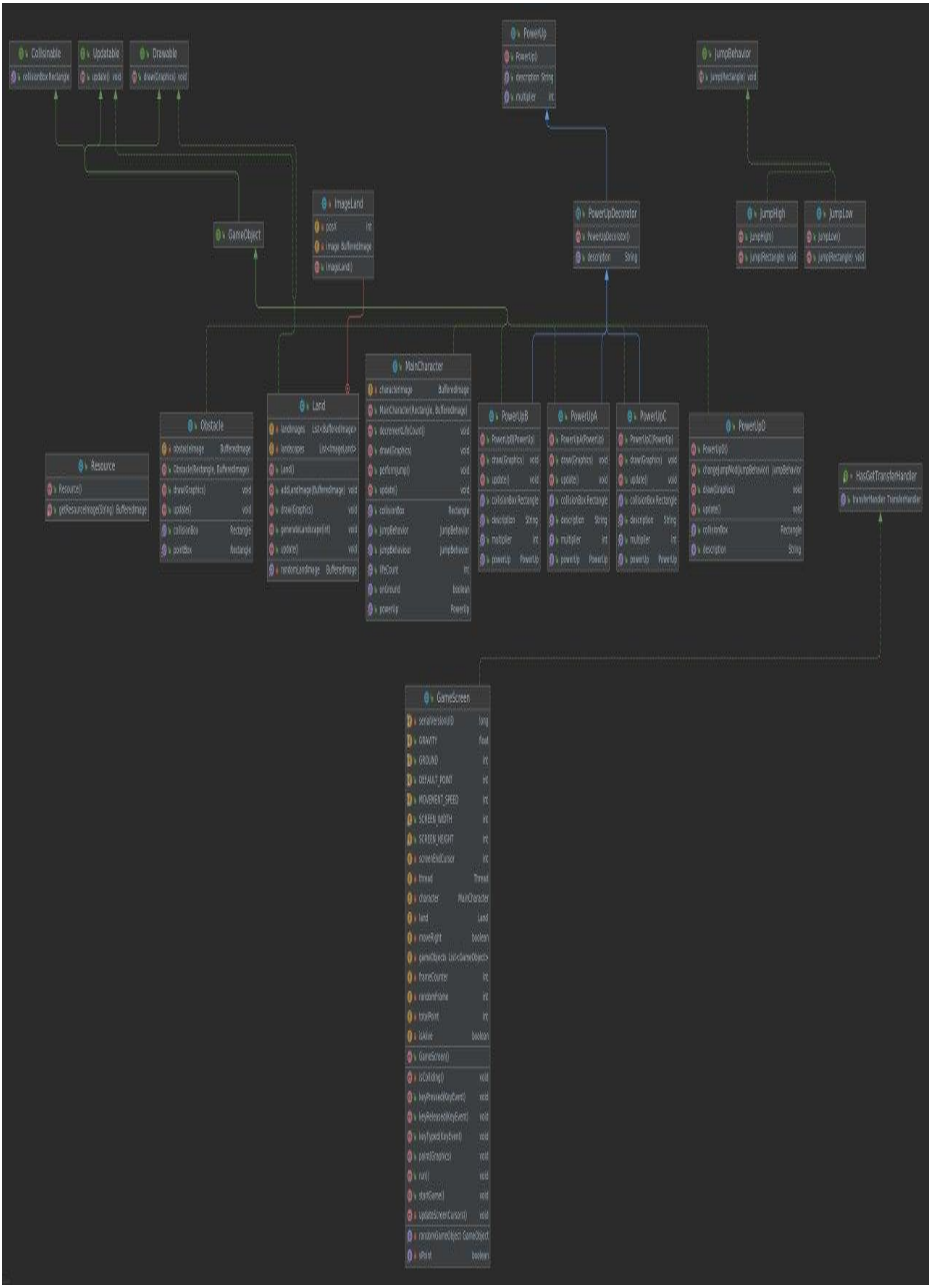
Green rectangle means powerup A, yellow rectangle means powerup B, red rectangle means powerup C, orange rectangle means powerup D.

Cactus images are obstacles.

Dino image is character.

At the top left corner, life of character, points of character and multiplier of character (as x 1) drawed.

I got help from a tutorial to learn design a game and pictures used are from that tutorial.


Collisinable
collisionBox Rectangle
Updatable
update() void
Drawable
draw(Graphics) void
GameObject
ImageLand
posX int
image BufferedImage
ImageLand()
PowerUp
PowerUp()
description String
multiplier int
JumpBehavior
jump(Rectangle) void
PowerUpDecorator
PowerUpDecorator()
description String
JumpHigh
JumpHigh()
jump(Rectangle) void
JumpLow
JumpLow()
jump(Rectangle) void
Resource
Resource()
getResourceImage(String) BufferedImage
Obstacle
obstacleImage BufferedImage
Obstacle(Rectangle, BufferedImage)
draw(Graphics) void
update() void
collisionBox Rectangle
pointBox Rectangle
Land
landImages List<BufferedImage>
landscapes List<ImageLand>
Land()
addLandImage(BufferedImage) void
draw(Graphics) void
generateLandscape(int) void
update() void
randomLandImage BufferedImage
MainCharacter
characterImage BufferedImage
MainCharacter(Rectangle, BufferedImage)
decrementLifeCount() void
draw(Graphics) void
performJump() void
update() void
collisionBox Rectangle
jumpBehavior JumpBehavior
jumpBehaviour JumpBehavior
lifeCount int
onGround boolean
powerUp PowerUp
PowerUpB
PowerUpB(PowerUp)
draw(Graphics) void
update() void
collisionBox Rectangle
description String
multiplier int
powerUp PowerUp
PowerUpA
PowerUpA(PowerUp)
draw(Graphics) void
update() void
collisionBox Rectangle
description String
multiplier int
powerUp PowerUp
PowerUpC
PowerUpC(PowerUp)
draw(Graphics) void
update() void
collisionBox Rectangle
description String
multiplier int
powerUp PowerUp
PowerUpD
PowerUpD()
changeJumpMod(JumpBehavior) JumpBehavior
draw(Graphics) void
update() void
collisionBox Rectangle
description String
HasGetTransferHandler
transferHandler TransferHandler
GameScreen
serialVersionUID long
GRAVITY float
GROUND int
DEFAULT_POINT int
MOVEMENT_SPEED int
SCREEN_WIDTH int
SCREEN_HEIGHT int
screenEndCursor int
thread Thread
character MainCharacter
land Land
moveRight boolean
gameObjects List<GameObject>
frameCounter int
randomFrame int
totalPoint int
isAlive boolean
GameScreen()
isColliding() void
keyPressed(KeyEvent) void
keyReleased(KeyEvent) void
keyTyped(KeyEvent) void
paint(Graphics) void
run() void
startGame() void
updateScreenCursors() void
randomGameObject GameObject
isPoint boolean