Pixel Saga Documentation

Created by

Napongthorn Charoenlap	**6631313321**
Thanagorn Chaiyut	**6631321321**

2110215 Programming Methodology
Semester 2 Year 2024
Chulalongkorn University

Pixel Saga Details

1. Introduction

Pixel Saga is a rogued-like platform game. In this game, you will be a wanderer sent from the earth. Because the earth's atmosphere is so bad that you can hardly live, you have to explore this new planet and survive as long as possible. While exploring, there are many monsters trying to defeat you to protect their planet. The more you beat them, the stronger you are and the more resources you have. And if you are defeated, you will be reborn but you will lose everything. Can you survive on this planet? Let's find out!

2. Gameplay

When you start playing this game, the first scene you will see is **The StartScene**.

Figure 1: The StartScene

In this scene, there will be three buttons to choose. The first is the play button that when you press this button. This button is used for playing this game. The second button is the option button. This button makes you go to **The OptionScene**. And you can adjust the sound of this game before playing. And the last button is the quit button. When you press this button, you will quit the game

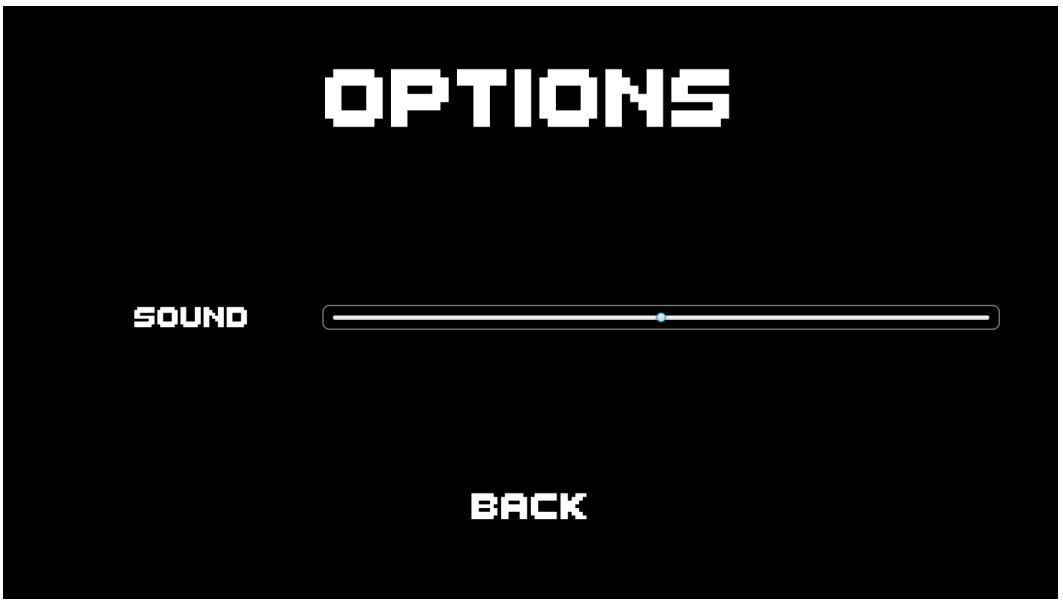


Figure 2: The OptionScene

In this scene, you can adjust the sound of the game by using this slider. If you are satisfied with the quality of the sound, you can press the back button to go back to The StartScene.

And when you press the start button in The StartScene, you will go to **The ClassSelectionScene**

Figure 3: The ClassSelectionScene

In this scene, you will spawn at the left of the map as a man wearing a white shirt. At the bottom left of the scene, there is a status bar to show the amount of hp you have and the icon of your attacks (after choosing a class). When you walk to the right, you will see two things: **The Class Symbols** and **The Vending Machine**

Figure 4: The Class Symbols

At the left side, there are three symbols for three classes to use: **The BroadSword** who has high attack points and high defend points, **The Dagger** who has high critical rate and high critical damage, and **The Bow** who has high evade rate and can attack from a distance. When you walk to each symbol, the game will show a symbol's popup letting you know the summary of that class and you can

check the inventory and the stats by pressing Tab and the game will show the stat of the class you currently choose and the perks you obtain. You can bring the mouse to the perk icon to check the ability of the perk you obtain. When you press Tab again or using LMB, it will close the inventory (At the bottom right there is a menu button used for go back to The StartScene)

Note: you have to be one of these classes before going to the next scene

Figures 5: The Broadsword

Figures 6: The Dagger

Figures 7: The Bow

Furthermore, each class has a different attack style and has his own special skill to use. The Broadsword will swing his broadsword dealing area damage. The Dagger will use a two-handed weapon and can teleport to the nearest enemy. The Bow will use a bow and can charge then shoot an arrow that can attack multiple enemies at the same time.

Figure 8: The Vending Machine

At the right side, there is a vending machine that can give you a perk in this game. You can choose the perk you want by changing the class and activating this vending machine. Do these two actions until you get the class and the perk you want.

List of the perk in the game

- Berserk (Common perk)
- DiscountMaster (Rare)
- Extrovert (Uncommon)
- FatalAttack (Common)
- Fortify (Uncommon)
- Introvert (Common)
- JukeMaster (Common)
- LuckyMan (Rare)
- Perfectionist (Common)
- PrecisionStrike (Common)
- RapidFire (Common)
- Reinforced (Common)
- Thorn (Common)
- TreasureHunter (Rare)
- Undead (Uncommon)
- Vampirism (Uncommon)

You can check the ability of the perk by opening the inventory and bringing your mouse to the perk icon you want.

If you are ready, walk to the right and let's go to **The FightScene!**

Figure 9: The FightScene

When you enter this scene for the first time, you will spawn in a random type of FightScene. At the bottom right, there is a minimap to check all entities' position in the map and there is some level data to show you such as the coin you currently have, the monster remaining in the map and the number of waves. You will have to beat all monsters in the map. The monsters will spawn wave by wave and for the same spawn position, it can be a different monster. There are many FightScene to play and each FightScene has different spawn positions and details. When you beat a monster, you will have a chance of getting **HPDropSmall** that can restore your hp for 5%.

Figure 10: HPDropSmall

There are 5 types of monsters that can spawn in The FightScene: **PeasantGiGee**, **ArcherGiGee**, **KleeGiGee**, **ThrowerGiGee** and **IceThrowerGiGee** (Monster's pictures are arranged consequently).

Figure 11: All Monsters in The FightScene

Some of the FightScene have a **Shroom**. When you jump on it, it will bounce you making your jump higher than your normal jump

Figure 12: Shroom

After you beat all the monsters, you will receive a reward based on the type of the FightScene and there are 4 types of FightScene in the game: **HPRoom, CoinRoom, PerkRoom and UpgradeRoom**

The HPRoom gives you a **HPReward Object** that can increase 25% maxHp then heal 50%

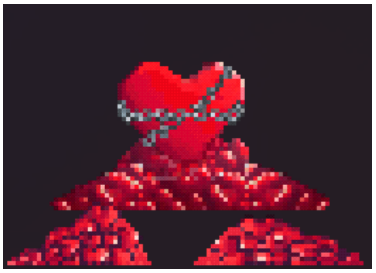


Figure 13: HPReward

The CoinRoom gives you a **CoinReward Object** that increases your coins by 200 coins

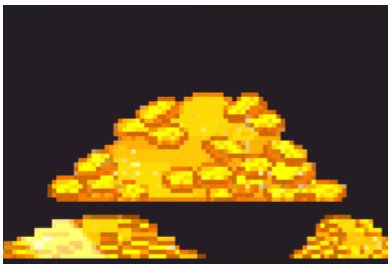


Figure 14: CoinReward

The PerkRoom gives you a **PerkChestRewerd Object** that lets you choose a perk from three random perks that you do not obtain.

Figure 15: PerkChestReward

The UpgradeRoom gives you a **Blacksmith Object** that lets you upgrade a perk (only one perk) that you obtain.

Figure 16: BlackSmith

After that, the opened doors will replace to the closed doors letting you choose the next type of The FightScene you want (Pictures at the left column are the open doors while the right are the closed doors)

Figure 17: HPRoom Doors

Figure 18: CoinRoom Doors

Figure 19: PerkRoom Doors

Figure 20: UpgradeRoom Doors

After you clear 5 FightScenes, the door to The MarketScene will appear. you have to go to **The MarketScene** to buy perks and upgrade your perks

Figure 21: MarketScene Doors

Figure 22: The MarketScene

When you reach this scene, there are the coins you have at the top right and in this scene, there are 3 things you can interact with. The first thing is **The Merchant** that sells you a potion that can restore 40% of your hp (you can buy the potion only one time and you can't buy it if your hp is full).

Figure 23: The Merchant

Next is **The Perk Area** where you can buy the perks that you don't obtain. This scene will generate four random perks you can buy (you can obtain only 6 perks!)

Figure 24: The Perk Area

And the last part is **The Blacksmith**. There are two blacksmiths in the game (which are the same mechanics) This area lets you upgrade your obtained perks from Tier1 to Tier2

Figure 25: The Blacksmith

The blacksmiths in The UpgradeRoom and The MarketScene are the same. When you interact with the blacksmith, it will create **The Upgrade Popup** letting you upgrade your perks

Figure 26: The Upgrade Popup

In this popup, there are two sections. At the left side, there are obtained perks and their current tier. When you click on a Tier 1 perk, it will create **The Upgrade Section** at the right side.

Figure 27: The Upgrade Section

There are two different Tier2s you can choose to upgrade, after picking a Tier2, your perk will be Tier2 perk and the upgrade section will disappear. To leave, clicking the back button at the bottom right

After preparing the perks, walking to the right and it's time to go to **The BossScene**

Figure 28: The BossScene

In this scene, you will fight against the boss of this game. The components in the same are the same as in The FightScene but there is an added component which is the boss health at the top middle. The boss is **BigBoy** who can attack using Flame, Stomping Attack and Shock Wave. When he dies, he will create a Nuclear Bomb for the last attack that deals massive damage and drop **HPLargeDrop** Object that can heal 40%. After clearing this scene, walk to the right and you will reach The FightScene again!

Figure 29: BigBoy

Figure 30: HPDropLarge

Figure 31: The DeathScene

If you die in the battle, this scene will appear. At the top of the screen, there is a red text that can encourage you (“Noob!”). At the middle, there is a restart button and if you press it, you will go back to The ClassSelectionScene and finally, the quot button for leaving this game

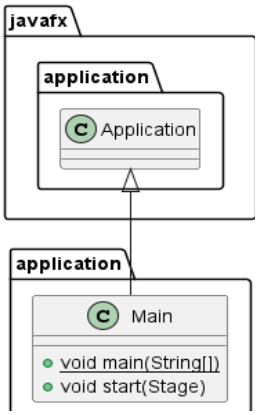
3. Key Controls

Key	Description
A	Move to the left
D	Move to the right
E	Interact
LMB	Normal Attack / Close the Inventory
RMB	Special Attack
Shift	Dash
Shift + Spacebar	Pass through the floating floor
Spacebar	Jump up
Tab	Check inventory

4. UML Diagrams

4.1. Animating

4.2. Application



4.3. BaseClass

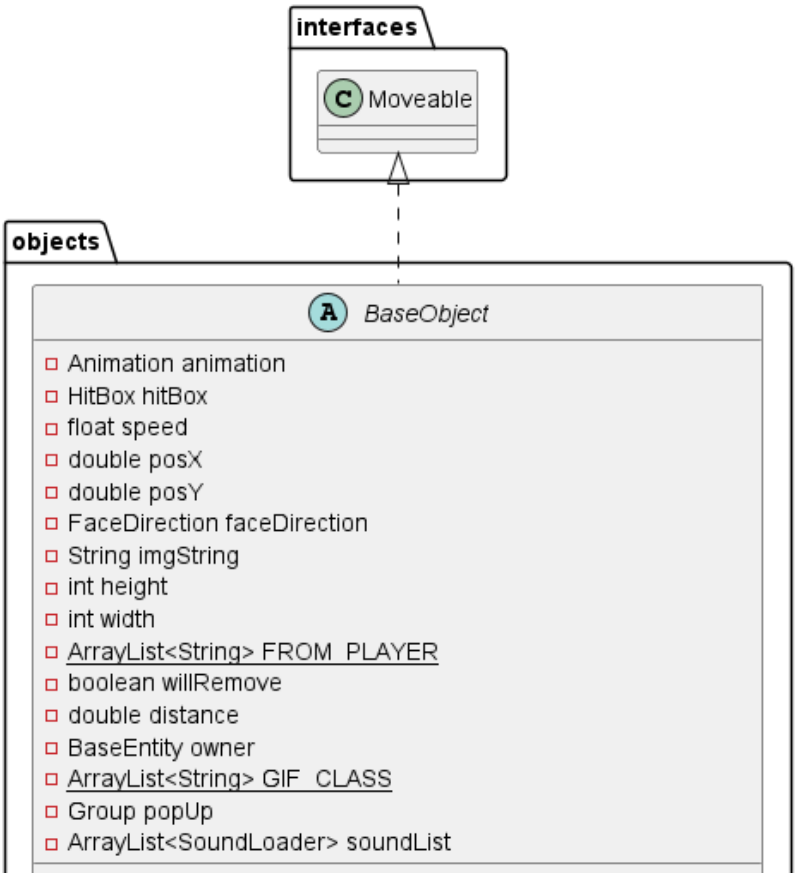
4.4. BaseEntity


void setCommonStatus()
void setUniqueStatus(double,double)
int getBaseMaxHp()
void setBaseMaxHp(int)
int getAddMaxHp()
void setAddMaxHp(int)
int getCurrentHp()
void setCurrentHp(int)
int getBaseAtk()
void setBaseAtk(int)
int getAddAtk()
void setAddAtk(int)
int getBaseDef()
void setBaseDef(int)
int getAddDef()
void setAddDef(int)
float getBaseMovementSpeed()
void setBaseMovementSpeed(float)
int getAddMovementSpeed()
void setAddMovementSpeed(int)
int getEvadeRate()
void setEvadeRate(int)
int getDamageIncrease()
void setDamageIncrease(int)
int getDamageDecrease()
void setDamageDecrease(int)
FaceDirection getFaceDirection()
void setFaceDirection(FaceDirection)
float getAttackRange()
void setAttackRange(float)
Animation getAnimation()
void setAnimation(Animation)
HitBox getHitBox()
void setHitBox(HitBox)
String getImgString()
void setImgString(String)
int getStandStillFrames()
void setStandStillFrames(int)
int getWalkFrames()
void setWalkFrames(int)
int getNormalAttackFrames()
void setNormalAttackFrames(int)
int getDeadFrames()
void setDeadFrames(int)
ArrayList<Integer> getFrameMakeDamageNormalAttack()
void setFrameMakeDamageNormalAttack(ArrayList<Integer>)
boolean canTakeDamage()
void setCanTakeDamage(boolean)
float getKnockbackDistance()
void setKnockbackDistance(float)
ArrayList<Integer> getDashFrame()
void setDashFrame(ArrayList<Integer>)
int getDashFrameCount()
void setDashFrameCount(int)
boolean isDash()
void setDash(boolean)
int getAnimationWidth()
void setAnimationWidth(int)
int getAnimationHeight()
void setAnimationHeight(int)
int getHitBoxWidth()
void setHitBoxWidth(int)
int getHitBoxHeight()
void setHitBoxHeight(int)
int getHitBoxXOffset()
void setHitBoxXOffset(int)
int getHitBoxYOffset()
void setHitBoxYOffset(int)
ArrayList<SoundLoader> getSoundList()
void setSoundList(ArrayList<SoundLoader>)
int getDamageReceived()
void setDamageReceived(int)

4.5. BaseMonster

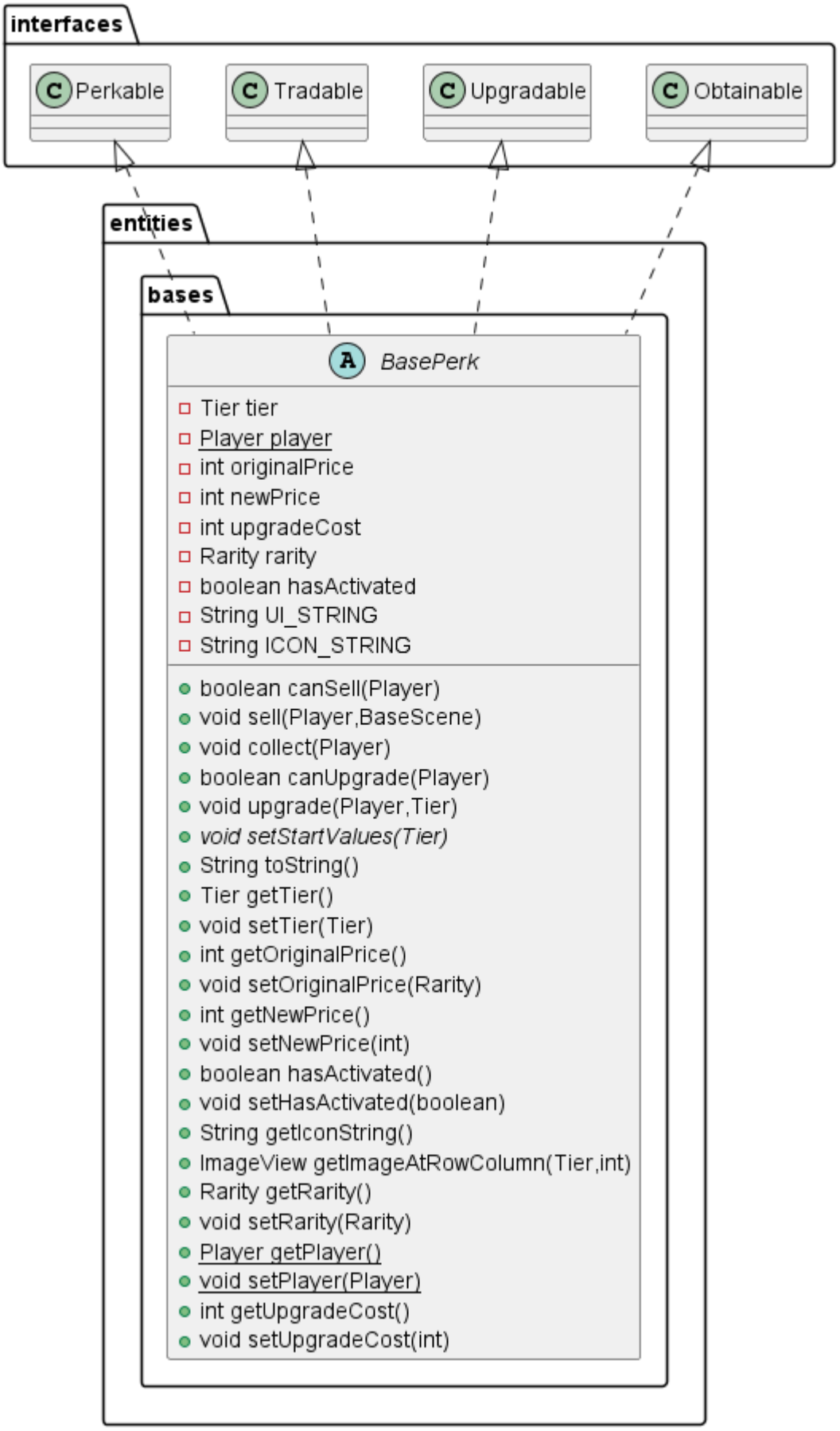

float getDeltaTilesX(Player)
float getDeltaTilesY(Player)
float getNormalAttackDelay()
void setNormalAttackDelay(float)
float getAddAttackDelay()
void setAddAttackDelay(float)
float getDropRate()
void setDropRate(float)
int getCoinDrop()
void setCoinDrop(int)
float getVisionRange()
void setVisionRange(float)
float getWalkTime()
void setWalkTime(float)
float getNewRandomTime()
float getTime()
void setTime(float)
float getStandTime()
void setStandTime(float)
void setWalk(boolean)
void setCanSee(boolean)
void setCanAttack(boolean)
boolean isPlayerAtRight(Player)
boolean isPlayerAtLeft(Player)
void setTurningTime(float)
float getTimeSinceLastAttack()
void setTimeSinceLastAttack(float)
int getKnockbackChance()
void setKnockbackChance(int)
boolean isSpawn()
void setSpawn(boolean)
FaceDirection getDashDirection()
void setDashDirection(FaceDirection)
void setMoreCoinDrop(boolean)
boolean isSlowdownByThornPerk()
void setSlowdownByThornPerk(boolean)
BaseBoss
float cooldownSkill1
float cooldownSkill2
float cooldownSkill3
float skillOneAttackRange
float skillTwoAttackRange
float skillThreeAttackRange
boolean canSkill1
ArrayList<Integer> skillToRandom
float attackDelay
ArrayList<Integer> frameMakeDdamageSkillOne
ArrayList<Integer> frameMakeDamageSkillTwo
ArrayList<Integer> frameMakeDamageSkillThree
«Create» BaseBoss()
void setCommonStatus()
float getCooldownSkill1()
void setCooldownSkill1(float)
float getCooldownSkill2()
void setCooldownSkill2(float)
float getCooldownSkill3()
void setCooldownSkill3(float)
float getSkillOneAttackRange()
void setSkillOneAttackRange(float)
float getSkillTwoAttackRange()
void setSkillTwoAttackRange(float)
float getSkillThreeAttackRange()
void setSkillThreeAttackRange(float)
ArrayList<Integer> getSkillToRandom()
boolean canSkill1()
void setCanSkill1(boolean)
boolean canSkill2()
void setCanSkill2(boolean)
boolean canSkill3()
void setCanSkill3(boolean)
float getAttackDelay()
void setAttackDelay(float)
ArrayList<Integer> getFrameMakeDamageSkillOne()
ArrayList<Integer> getFrameMakeDamageSkillTwo()
ArrayList<Integer> getFrameMakeDamageSkillThree()
BaseGiGee
«Create» BaseGiGee()
void setCommonStatus()

4.6. BaseObject

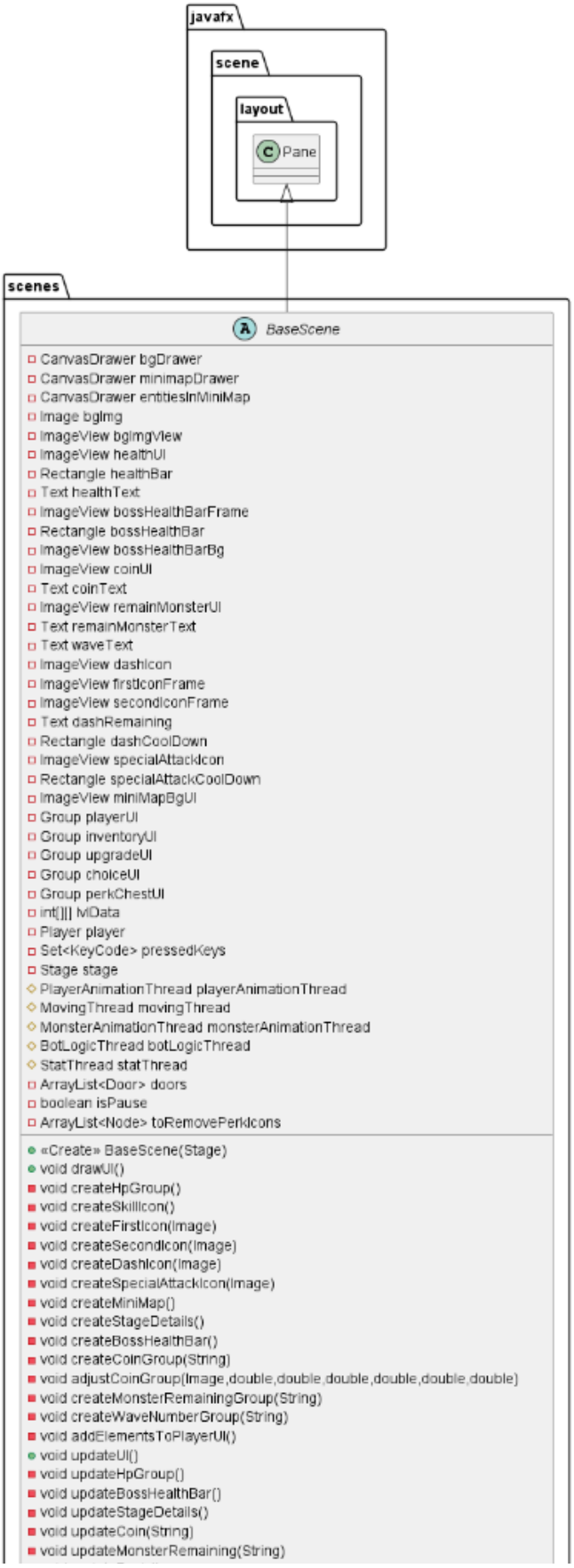



«Create» BaseObject(BaseEntity,double,double,FaceDirection)
void move(BaseScene)
void checkHitTile(double,BaseScene)
void moveObject(double)
void checkHitMonster(BaseScene)
void checkHitPlayer(BaseScene)
void makeDamage(Player,BaseScene)
void makeThrowerBombDamage(Player,BaseScene)
void applyIceThrowerSlow(Player)
void setSlowTimer(Player,int,int)
void heal(Player)
void activateSuperJump(Player,BaseScene)
void moveInsideInteractObject(BaseScene)
void moveOutInteractObject(BaseScene)
void checkFall(BaseScene)
void jump(BaseScene)
void createMerchantPopUp(Merchant,BaseScene)
void removePopUp(BaseScene)
void addClassSymbolPopUp(Symbol,BaseScene)
void createClassDetail(Symbol)
void addPerkSymbolPopUp(BasePerk,BaseScene)
void createPotionPrice(Merchant)
void createPerkPrice(BasePerk)
void createPerkDetail(BasePerk)
boolean isHitEntity(BaseEntity)
boolean checkEntityHitbox(double,BaseEntity)
void runOneTimeAnimation(int,int,BaseScene,boolean)
Animation getAnimation()
void setAnimation(Animation)
float getSpeed()
void setSpeed(float)
double getPosX()
void setPosX(double)
double getPosY()
void setPosY(double)
FaceDirection getFaceDirection()
void setFaceDirection(FaceDirection)
String getImgString()
void setImgString(String)
HitBox getHitBox()
void setHitBox(HitBox)
int getHeight()
void setHeight(int)
int getWidth()
void setWidth(int)
boolean willRemove()
void setWillRemove(boolean)
double getDistance()
void setDistance(double)
BaseEntity getOwner()
void setOwner(BaseEntity)
Group getPopUp()
ArrayList<SoundLoader> getSoundList()
void setSoundList(ArrayList<SoundLoader>)
void setCanIgnoreTile(boolean)
void setCanFall(boolean)
boolean isDamageDealt()
void setDamageDealt(boolean)
boolean canInteract()
void setCanInteract(boolean)

4.7 BasePerk

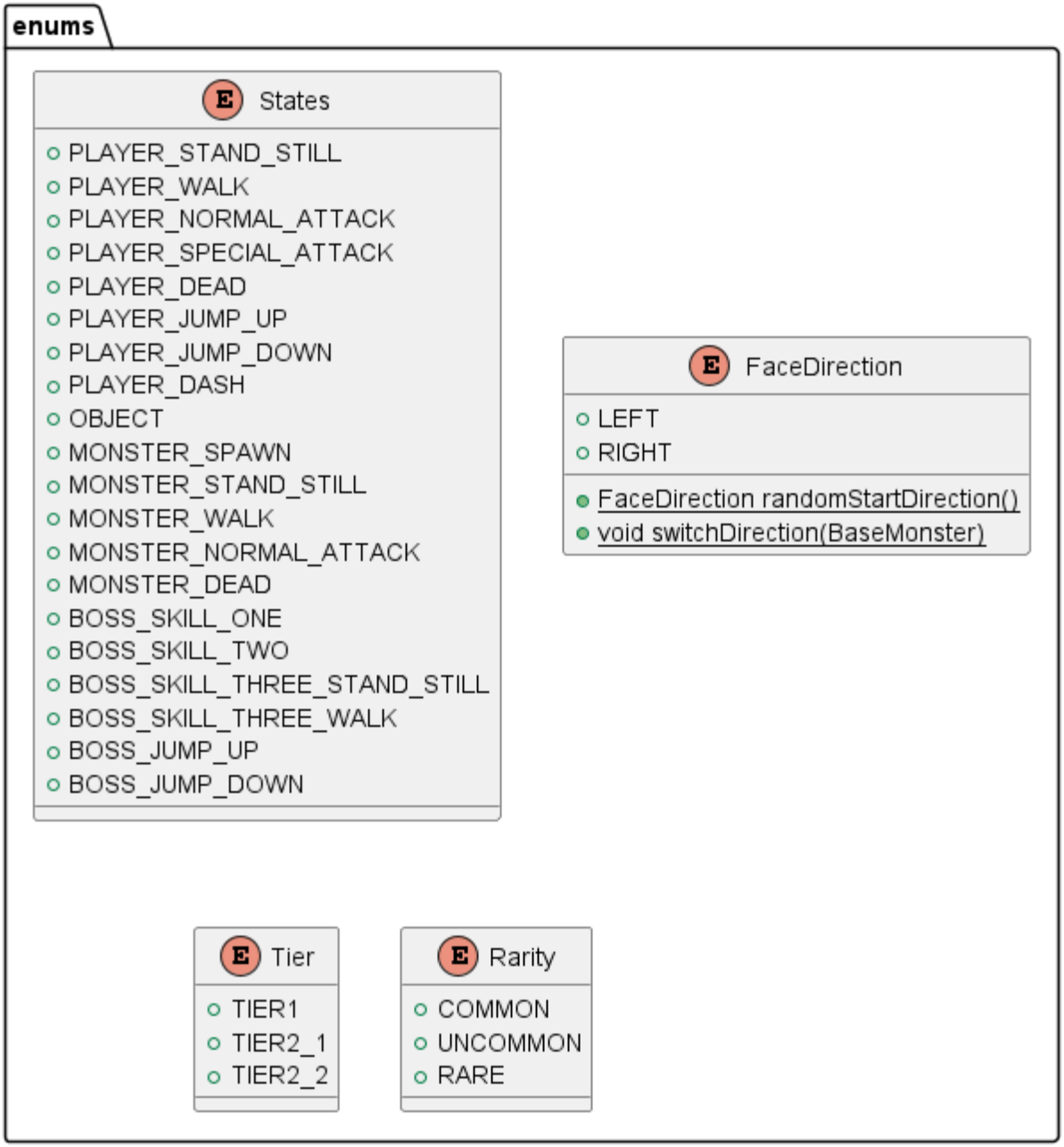

interfaces
Perkable
Tradable
Upgradable
Obtainable
entities
bases
BasePerk
Tier tier
Player player
int originalPrice
int newPrice
int upgradeCost
Rarity rarity
boolean hasActivated
String UI_STRING
String ICON_STRING
boolean canSell(Player)
void sell(Player,BaseScene)
void collect(Player)
boolean canUpgrade(Player)
void upgrade(Player,Tier)
void setStartValues(Tier)
String toString()
Tier getTier()
void setTier(Tier)
int getOriginalPrice()
void setOriginalPrice(Rarity)
int getNewPrice()
void setNewPrice(int)
boolean hasActivated()
void setHasActivated(boolean)
String getIconString()
ImageView getImageAtRowColumn(Tier,int)
Rarity getRarity()
void setRarity(Rarity)
Player getPlayer()
void setPlayer(Player)
int getUpgradeCost()
void setUpgradeCost(int)

4.8. BaseScenes

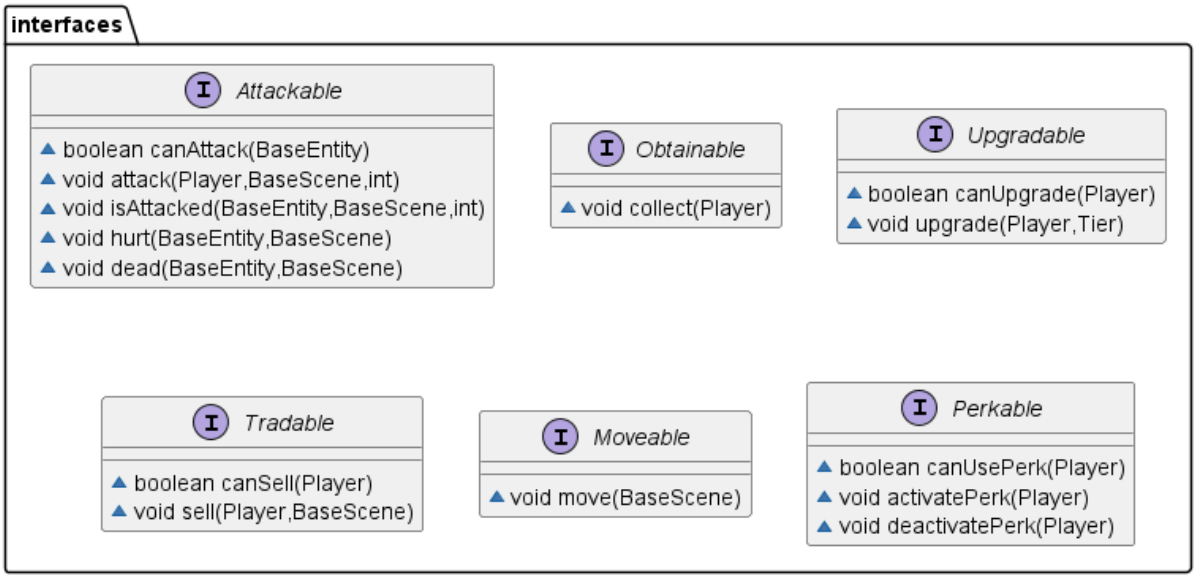



void updateDash()
void updateWaveNumber(String)
void updateMiniMap()
void drawBackground()
void createInventoryUI()
void drawAllPerks(Group,float[],float[])
void drawPerk(BasePerk,float,float,Group)
void createPerkText(BasePerk,float,float,Group)
void createPerkIcon(BasePerk,float,float,Group)
ImageView showMiniPopUp(BasePerk,double,double)
void pickPerkToUpgrade(BasePerk)
void createPerkChoiceToUpgrade(BasePerk,Tier,double,double)
void setUpChoice(ImageView,double,double,BasePerk,Tier)
void setUpPriceGroup(ImageView,BasePerk,Group)
Group createMenuButton()
void addAllStats(double,ImageView,Group)
void createUpgradeUI()
Group createBackButton()
void createPerkChestUI()
void setPerkChestBackground(ImageView)
void addSelectedPerksForChest(ArrayList<BasePerk>,ImageView,BaseScene)
void selectPerkToObtain(BasePerk,BaseScene)
void setKey(Animation)
void handleKeyPressed(Animation)
void interact()
void interactInClassSelectionScene()
void interactInFightScene()
void interactInMarketScene()
void pickPerkToBuy(Symbol)
void removePerk(Symbol)
void activatePassingTheFloor()
void activateJump(Animation)
void activateDash(Animation)
void obtainBaseReward()
void obtainPerkChestReward()
void pressInventory()
void nextStage()
void interactSymbol(Symbol)
void pressMoveLeft(Animation)
void pressMoveRight(Animation)
void createOpenDoor()
void selectClass(String)
void setPlayerClass(String)
void setPlayerPosition(String)
void setFaceDirection(FaceDirection)
void startAllThreads(BaseScene)
void handleKeyReleased(KeyCode,Animation)
void releaseMoveRight(Animation)
void releaseMoveLeft(Animation)
void handleMouseClicked(MouseButton,Animation)
void activateNormalAttack(Animation)
void activateSpecialAttack(Animation)
void handleMousePressed(MouseButton,Animation)
void activateBowSpecialAttack(Animation)
void activateBroadSwordSpecialAttack(Animation)
void activateDaggerSpecialAttack(Animation)
void handleMouseReleased(MouseButton,Animation)
void blink(ImageView)
void setScenePosition()
String toString()
void stopAllThreads()
void setLowBrightness()
void setDefaultBrightness()
Player getPlayer()
void setPlayer(Player)
Image getBgImg()
void setBgImg(Image)
ImageView getBgImgView()
void setBgImgView(ImageView)
Set<KeyCode> getPressedKeys()
int[][] getLvlData()
void setLvlData(int[][])
Stage getStage()
Group getPlayerUI()
ArrayList<Door> getDoors()
void setDoors(ArrayList<Door>)
boolean isPause()
void setPause(boolean)
ImageView getDashIcon()
Rectangle getDashCoolDown()
ImageView getSpecialAttackIcon()
Rectangle getSpecialAttackCoolDown()

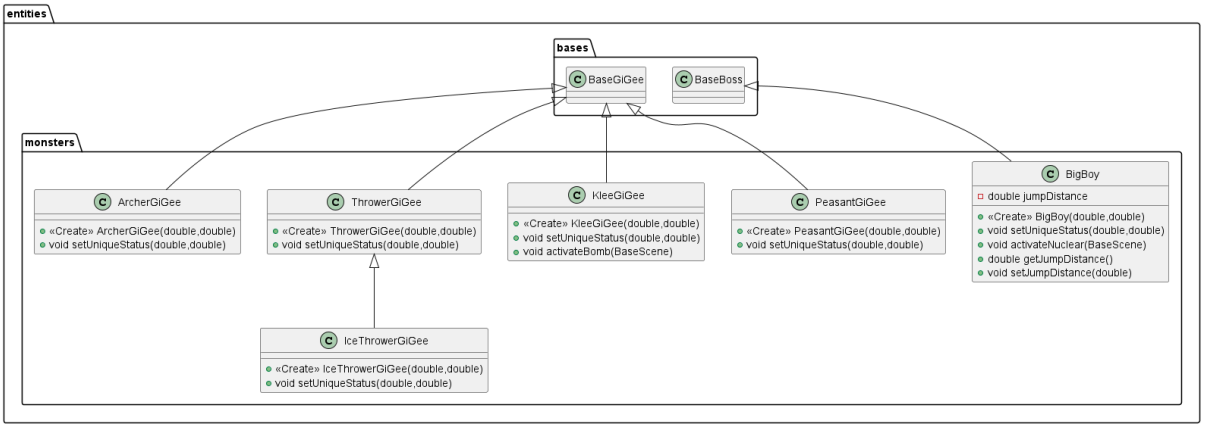
4.9. Enums


enums
States
PLAYER_STAND_STILL
PLAYER_WALK
PLAYER_NORMAL_ATTACK
PLAYER_SPECIAL_ATTACK
PLAYER_DEAD
PLAYER_JUMP_UP
PLAYER_JUMP_DOWN
PLAYER_DASH
OBJECT
MONSTER_SPAWN
MONSTER_STAND_STILL
MONSTER_WALK
MONSTER_NORMAL_ATTACK
MONSTER_DEAD
BOSS_SKILL_ONE
BOSS_SKILL_TWO
BOSS_SKILL_THREE_STAND_STILL
BOSS_SKILL_THREE_WALK
BOSS_JUMP_UP
BOSS_JUMP_DOWN
FaceDirection
LEFT
RIGHT
FaceDirection randomStartDirection()
void switchDirection(BaseMonster)
Tier
TIER1
TIER2_1
TIER2_2
Rarity
COMMON
UNCOMMON
RARE

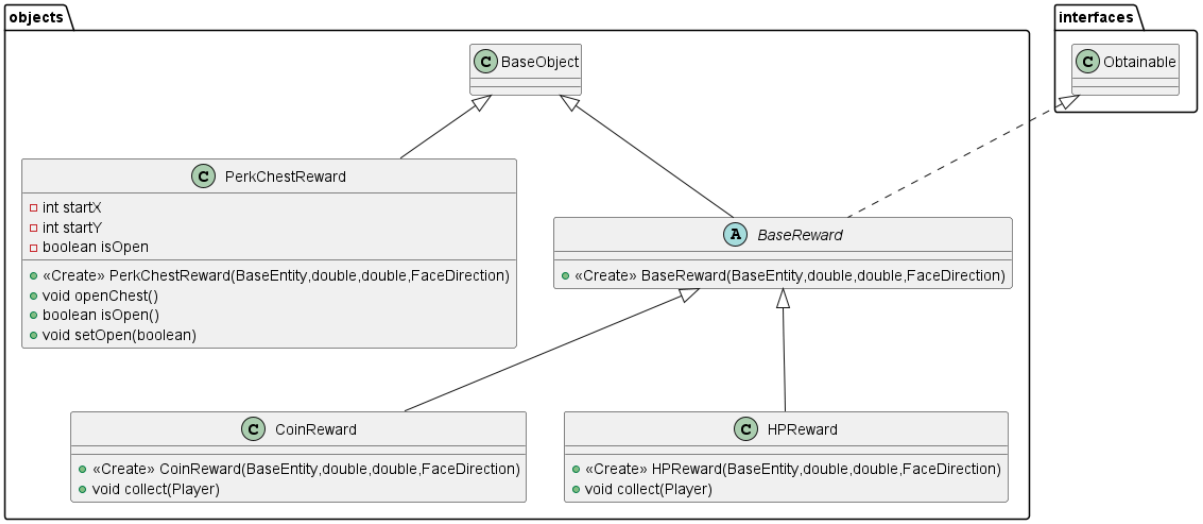
4.10. Interfaces

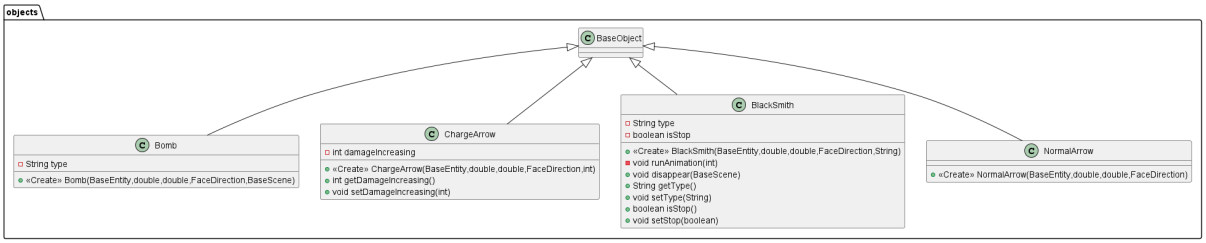


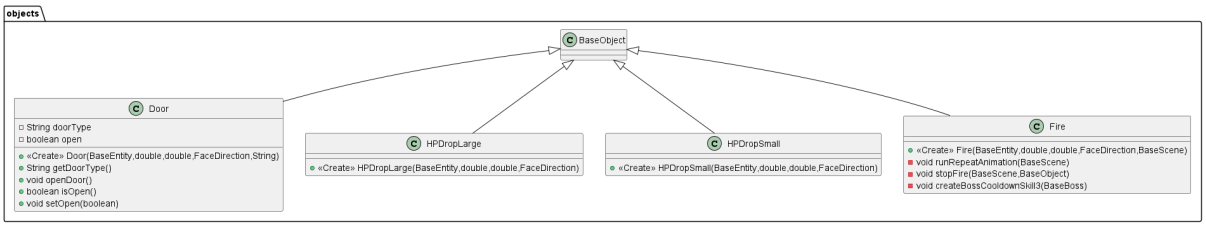
4.11. Monsters

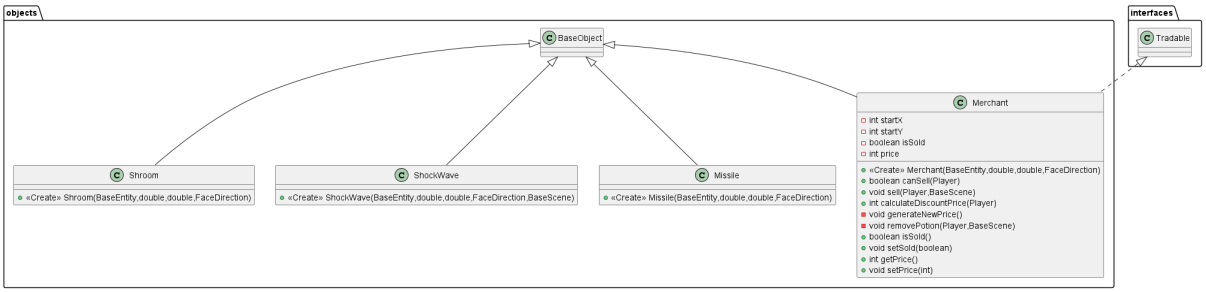


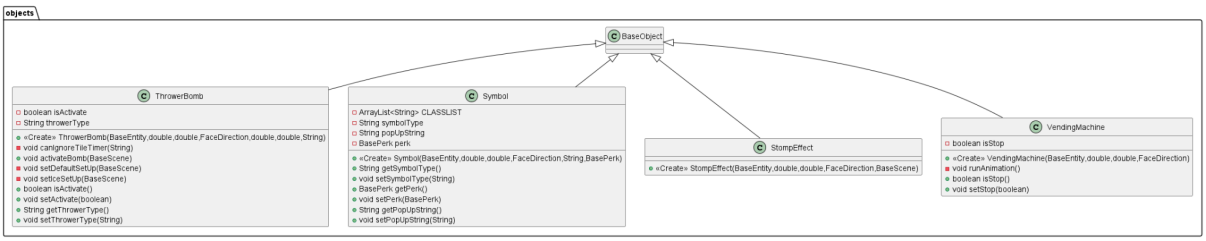
4.12. Objects


objects
interfaces
BaseObject
Obtainable
PerkChestReward
int startX
int startY
boolean isOpen
«Create» PerkChestReward(BaseEntity,double,double,FaceDirection)
void openChest()
boolean isOpen()
void setOpen(boolean)
BaseReward
«Create» BaseReward(BaseEntity,double,double,FaceDirection)
CoinReward
«Create» CoinReward(BaseEntity,double,double,FaceDirection)
void collect(Player)
HPReward
«Create» HPReward(BaseEntity,double,double,FaceDirection)
void collect(Player)

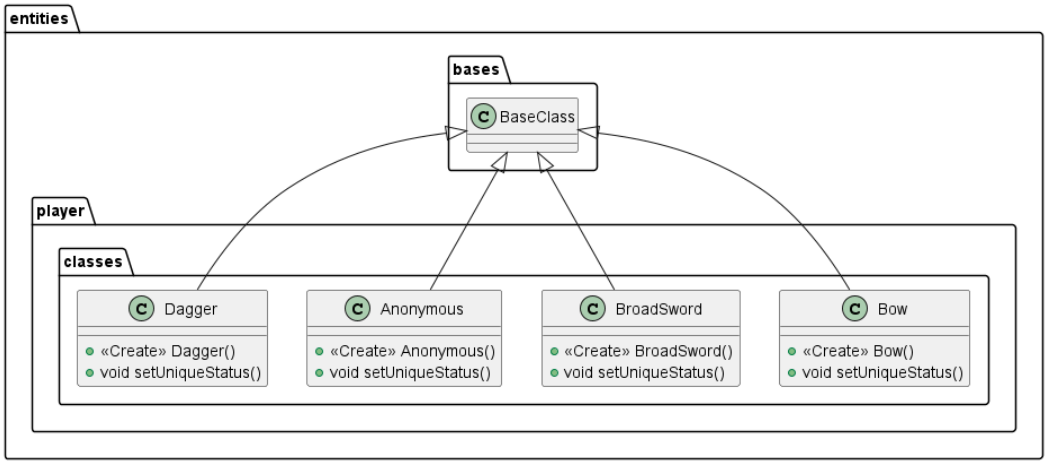

objects
BaseObject
Bomb
String type
«Create» Bomb(BaseEntity,double,double,FaceDirection,BaseScene)
ChargeArrow
int damageIncreasing
«Create» ChargeArrow(BaseEntity,double,double,FaceDirection,int)
int getDamageIncreasing()
void setDamageIncreasing(int)
BlackSmith
String type
boolean isStop
«Create» BlackSmith(BaseEntity,double,double,FaceDirection,String)
void runAnimation(int)
void disappear(BaseScene)
String getType()
void setType(String)
boolean isStop()
void setStop(boolean)
NormalArrow
«Create» NormalArrow(BaseEntity,double,double,FaceDirection)


objects
BaseObject
Door
String doorType
boolean open
«Create» Door(BaseEntity,double,double,FaceDirection,String)
String getDoorType()
void openDoor()
boolean isOpen()
void setOpen(boolean)
HPDropLarge
«Create» HPDropLarge(BaseEntity,double,double,FaceDirection)
HPDropSmall
«Create» HPDropSmall(BaseEntity,double,double,FaceDirection)
Fire
«Create» Fire(BaseEntity,double,double,FaceDirection,BaseScene)
void runRepeatAnimation(BaseScene)
void stopFire(BaseScene,BaseObject)
void createBossCooldownSkill3(BaseBoss)

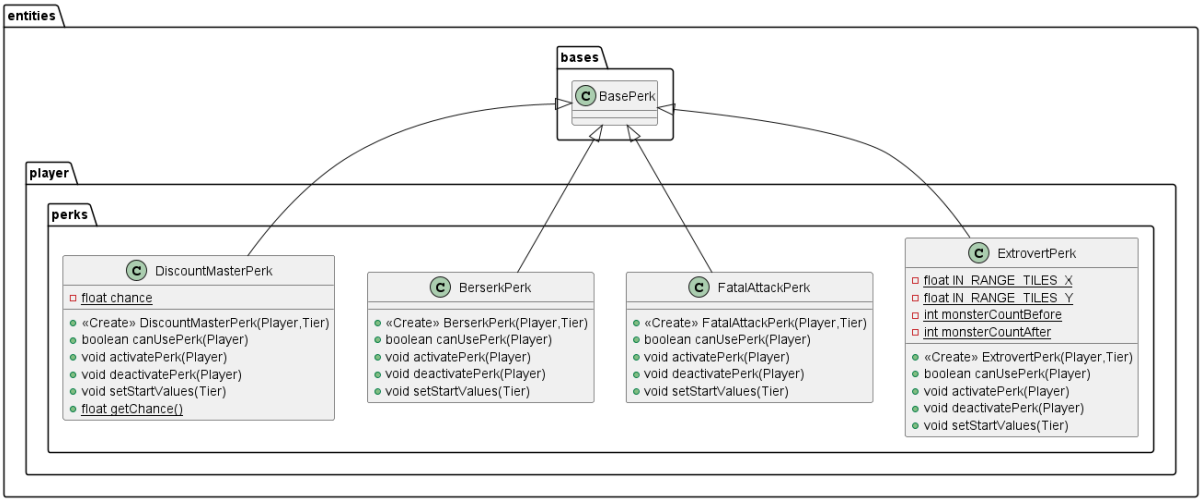

objects
interfaces
BaseObject
Tradable
Merchant
int startX
int startY
boolean isSold
int price
«Create» Merchant(BaseEntity,double,double,FaceDirection)
boolean canSell(Player)
void sell(Player,BaseScene)
int calculateDiscountPrice(Player)
void generateNewPrice()
void removePotion(Player,BaseScene)
boolean isSold()
void setSold(boolean)
int getPrice()
void setPrice(int)
Shroom
«Create» Shroom(BaseEntity,double,double,FaceDirection)
ShockWave
«Create» ShockWave(BaseEntity,double,double,FaceDirection,BaseScene)
Missile
«Create» Missile(BaseEntity,double,double,FaceDirection)

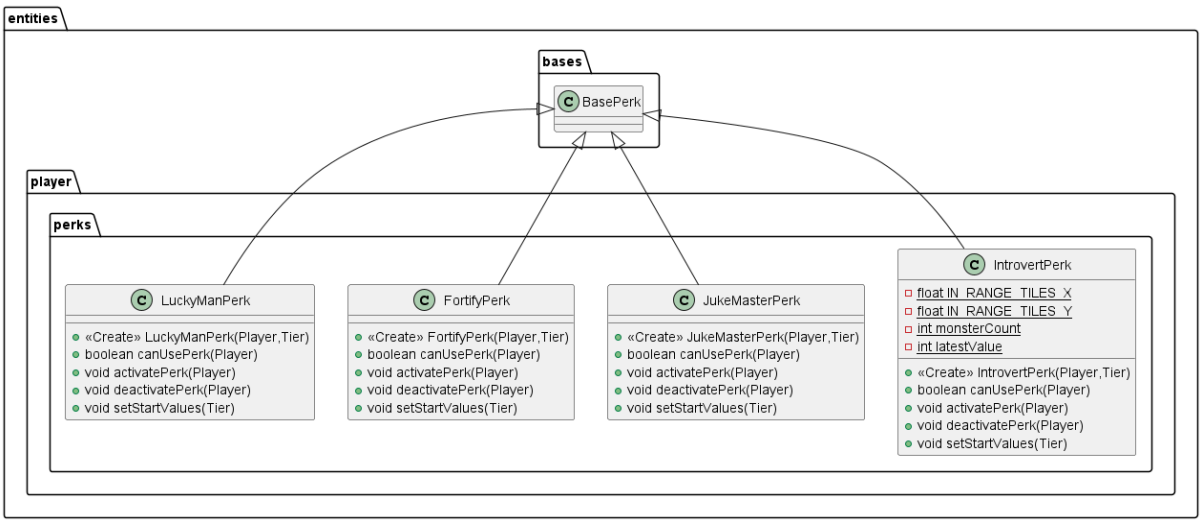

objects
BaseObject
ThrowerBomb
boolean isActivate
String throwerType
«Create» ThrowerBomb(BaseEntity,double,double,FaceDirection,double,double,String)
void canIgnoreTileTimer(String)
void activateBomb(BaseScene)
void setDefaultSetUp(BaseScene)
void setIceSetUp(BaseScene)
boolean isActivate()
void setActivate(boolean)
String getThrowerType()
void setThrowerType(String)
Symbol
ArrayList<String> CLASSLIST
String symbolType
String popUpString
BasePerk perk
«Create» Symbol(BaseEntity,double,double,FaceDirection,String,BasePerk)
String getSymbolType()
void setSymbolType(String)
BasePerk getPerk()
void setPerk(BasePerk)
String getPopUpString()
void setPopUpString(String)
StompEffect
«Create» StompEffect(BaseEntity,double,double,FaceDirection,BaseScene)
VendingMachine
boolean isStop
«Create» VendingMachine(BaseEntity,double,double,FaceDirection)
void runAnimation()
boolean isStop()
void setStop(boolean)

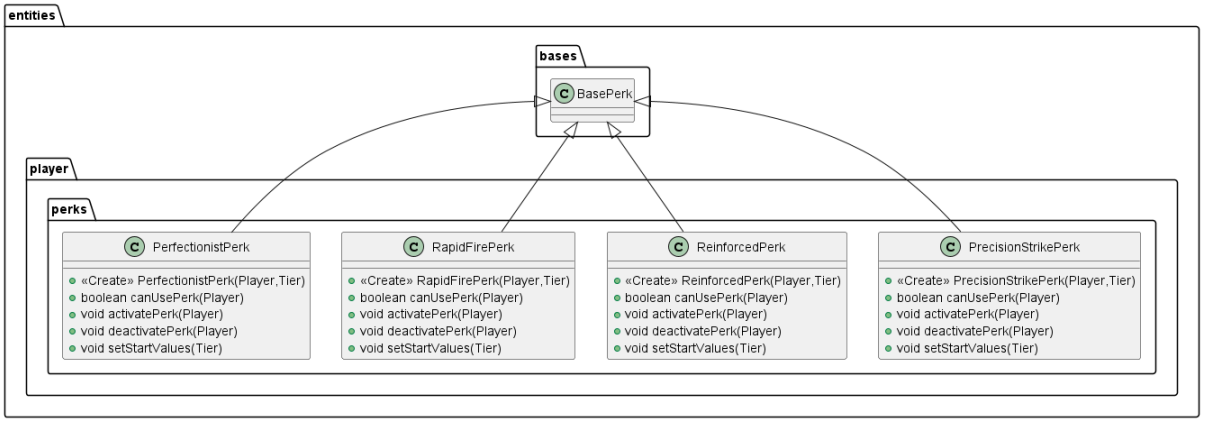
4.13. PlayerClasses

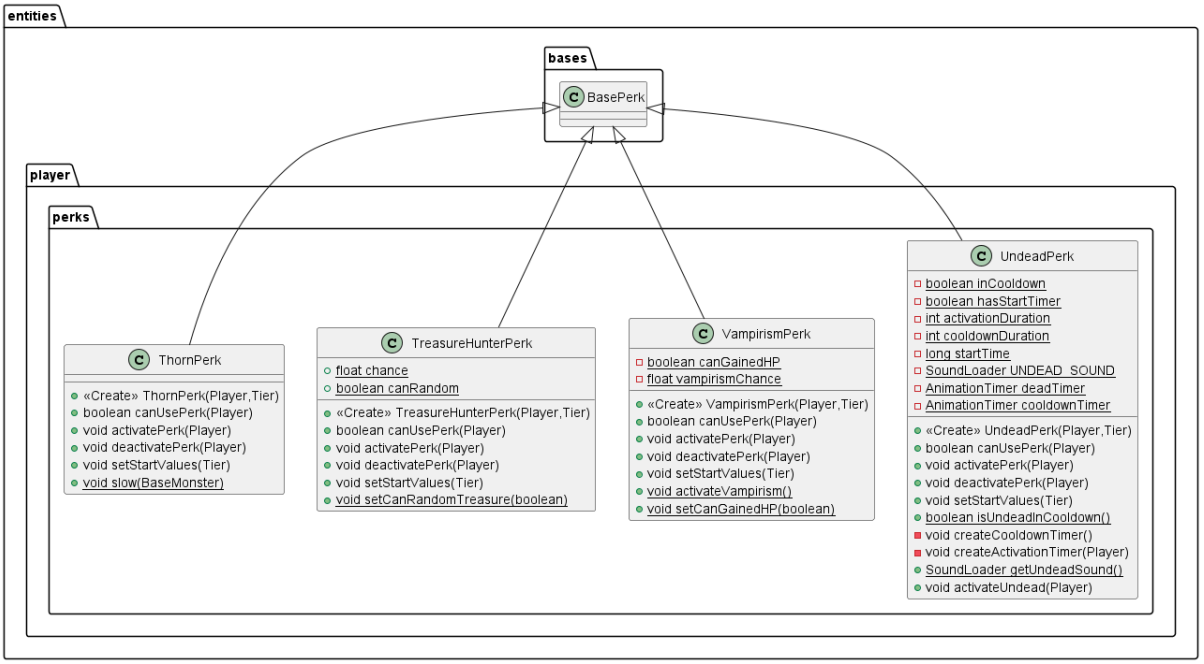


4.14. PlayerPerks

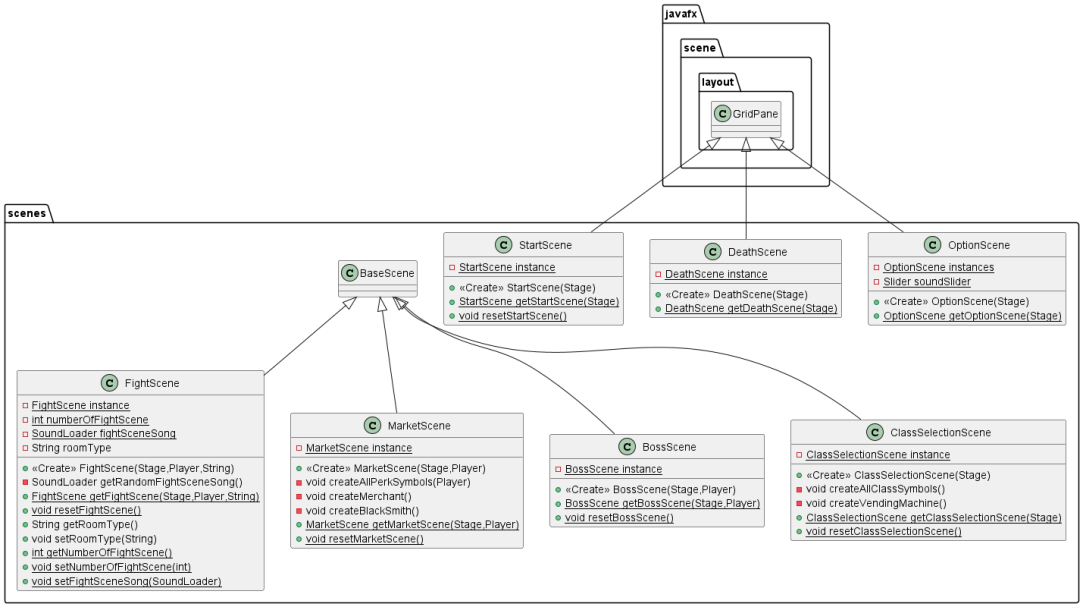




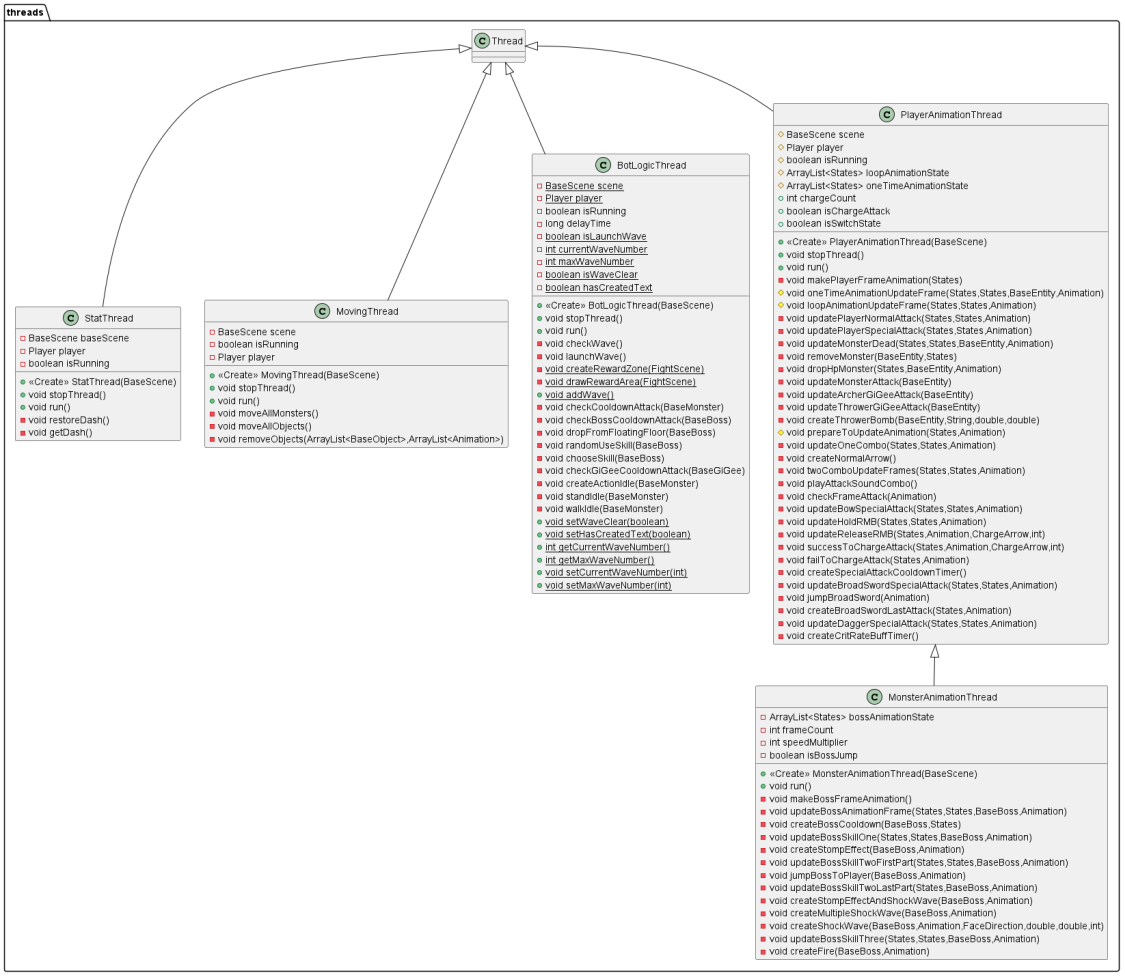

entities
bases
BasePerk
player
perks
PerfectionistPerk
«Create» PerfectionistPerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
RapidFirePerk
«Create» RapidFirePerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
ReinforcedPerk
«Create» ReinforcedPerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
PrecisionStrikePerk
«Create» PrecisionStrikePerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)


entities
bases
BasePerk
player
perks
ThornPerk
«Create» ThornPerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
void slow(BaseMonster)
TreasureHunterPerk
float chance
boolean canRandom
«Create» TreasureHunterPerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
void setCanRandomTreasure(boolean)
VampirismPerk
boolean canGainedHP
float vampirismChance
«Create» VampirismPerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
void activateVampirism()
void setCanGainedHP(boolean)
UndeadPerk
boolean inCooldown
boolean hasStartTimer
int activationDuration
int cooldownDuration
long startTime
SoundLoader UNDEAD_SOUND
AnimationTimer deadTimer
AnimationTimer cooldownTimer
«Create» UndeadPerk(Player,Tier)
boolean canUsePerk(Player)
void activatePerk(Player)
void deactivatePerk(Player)
void setStartValues(Tier)
boolean isUndeadInCooldown()
void createCooldownTimer()
void createActivationTimer(Player)
SoundLoader getUndeadSound()
void activateUndead(Player)

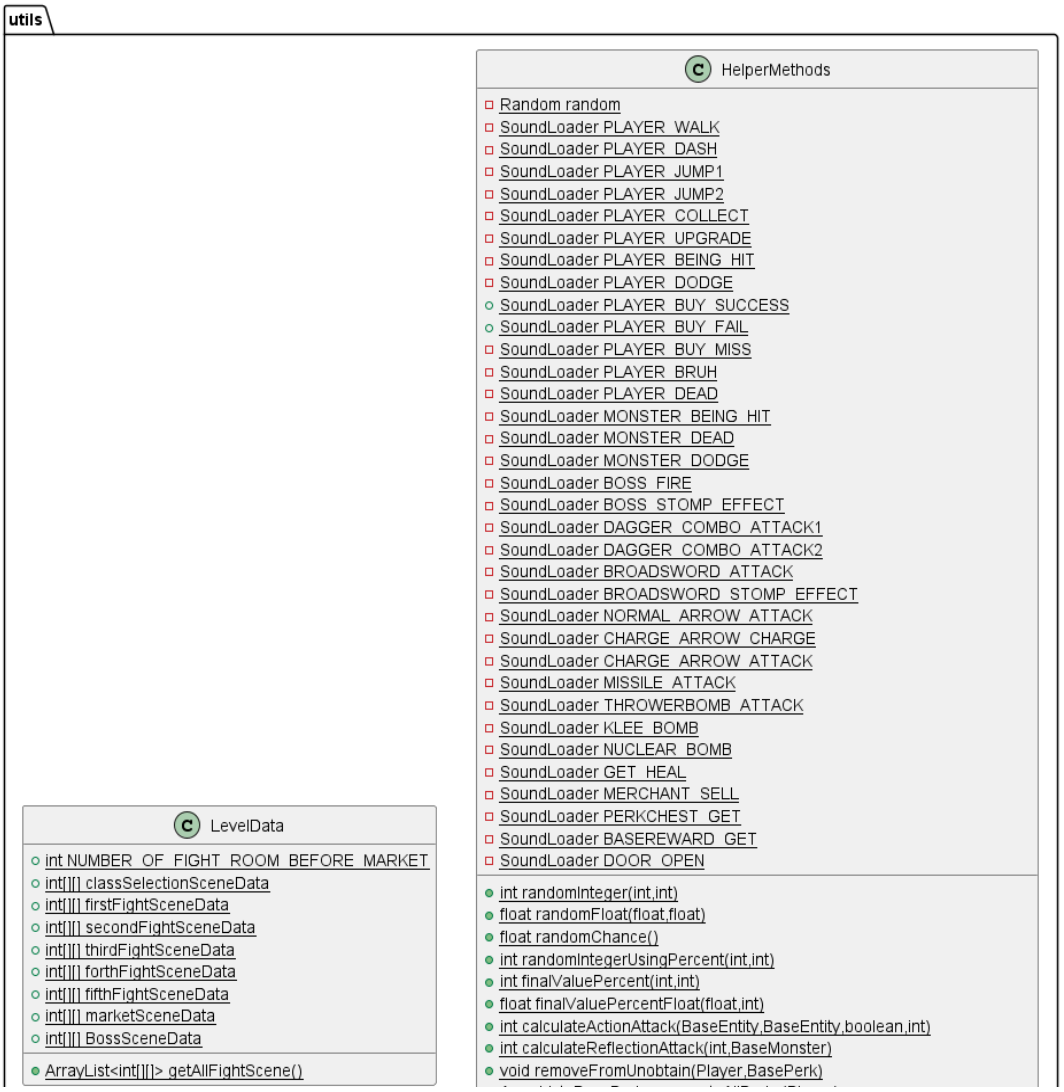
4.15. Scenes


javafx
scene
layout
GridPane
scenes
BaseScene
StartScene
StartScene instance
«Create» StartScene(Stage)
StartScene getStartScene(Stage)
void resetStartScene()
DeathScene
DeathScene instance
«Create» DeathScene(Stage)
DeathScene getDeathScene(Stage)
OptionScene
OptionScene instances
Slider soundSlider
«Create» OptionScene(Stage)
OptionScene getOptionScene(Stage)
FightScene
FightScene instance
int numberOfFightScene
SoundLoader fightSceneSong
String roomType
«Create» FightScene(Stage,Player,String)
SoundLoader getRandomFightSceneSong()
FightScene getFightScene(Stage,Player,String)
void resetFightScene()
String getRoomType()
void setRoomType(String)
int getNumberOfFightScene()
void setNumberOfFightScene(int)
void setFightSceneSong(SoundLoader)
MarketScene
MarketScene instance
«Create» MarketScene(Stage,Player)
void createAllPerkSymbols(Player)
void createMerchant()
void createBlackSmith()
MarketScene getMarketScene(Stage,Player)
void resetMarketScene()
BossScene
BossScene instance
«Create» BossScene(Stage,Player)
BossScene getBossScene(Stage,Player)
void resetBossScene()
ClassSelectionScene
ClassSelectionScene instance
«Create» ClassSelectionScene(Stage)
void createAllClassSymbols()
void createVendingMachine()
ClassSelectionScene getClassSelectionScene(Stage)
void resetClassSelectionScene()

4.16. Threads

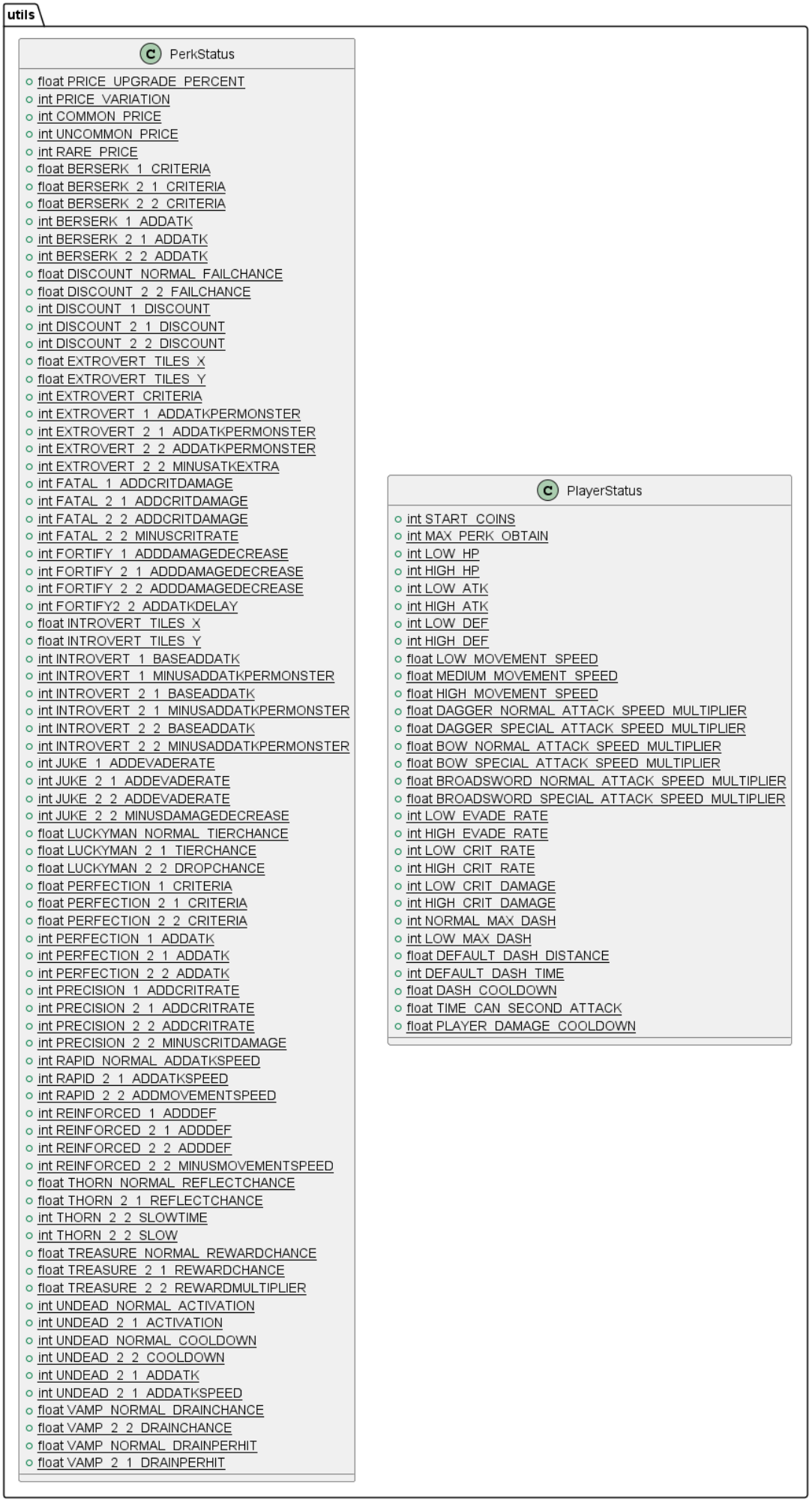

threads
Thread
StatThread
- BaseScene baseScene
- Player player
- boolean isRunning
+ «Create» StatThread(BaseScene)
+ void stopThread()
+ void run()
- void restoreDash()
- void getDash()
MovingThread
- BaseScene scene
- boolean isRunning
- Player player
+ «Create» MovingThread(BaseScene)
+ void stopThread()
+ void run()
- void moveAllMonsters()
- void moveAllObjects()
- void removeObjects(ArrayList<BaseObject>,ArrayList<Animation>)
BotLogicThread
- BaseScene scene
- Player player
- boolean isRunning
- long delayTime
- boolean isLaunchWave
- int currentWaveNumber
- int maxWaveNumber
- boolean isWaveClear
- boolean hasCreatedText
+ «Create» BotLogicThread(BaseScene)
+ void stopThread()
+ void run()
- void checkWave()
- void launchWave()
- void createRewardZone(FightScene)
- void drawRewardArea(FightScene)
+ void addWave()
- void checkCooldownAttack(BaseMonster)
- void checkBossCooldownAttack(BaseBoss)
- void dropFromFloatingFloor(BaseBoss)
- void randomUseSkill(BaseBoss)
- void chooseSkill(BaseBoss)
- void checkGiGeeCooldownAttack(BaseGiGee)
- void createActionIdle(BaseMonster)
- void standIdle(BaseMonster)
- void walkIdle(BaseMonster)
+ void setWaveClear(boolean)
+ void setHasCreatedText(boolean)
+ int getCurrentWaveNumber()
+ int getMaxWaveNumber()
+ void setCurrentWaveNumber(int)
+ void setMaxWaveNumber(int)
PlayerAnimationThread
◇ BaseScene scene
◇ Player player
◇ boolean isRunning
◇ ArrayList<States> loopAnimationState
◇ ArrayList<States> oneTimeAnimationState
○ int chargeCount
○ boolean isChargeAttack
○ boolean isSwitchState
+ «Create» PlayerAnimationThread(BaseScene)
+ void stopThread()
+ void run()
- void makePlayerFrameAnimation(States)
◇ void oneTimeAnimationUpdateFrame(States,States,BaseEntity,Animation)
◇ void loopAnimationUpdateFrame(States,States,Animation)
- void updatePlayerNormalAttack(States,States,Animation)
- void updatePlayerSpecialAttack(States,States,Animation)
- void updateMonsterDead(States,States,BaseEntity,Animation)
- void removeMonster(BaseEntity,States)
- void dropHpMonster(States,BaseEntity,Animation)
- void updateMonsterAttack(BaseEntity)
- void updateArcherGiGeeAttack(BaseEntity)
- void updateThrowerGiGeeAttack(BaseEntity)
- void createThrowerBomb(BaseEntity,String,double,double)
◇ void prepareToUpdateAnimation(States,Animation)
- void updateOneCombo(States,States,Animation)
- void createNormalArrow()
- void twoComboUpdateFrames(States,States,Animation)
- void playAttackSoundCombo()
- void checkFrameAttack(Animation)
- void updateBowSpecialAttack(States,States,Animation)
- void updateHoldRMB(States,States,Animation)
- void updateReleaseRMB(States,Animation,ChargeArrow,int)
- void successToChargeAttack(States,Animation,ChargeArrow,int)
- void failToChargeAttack(States,Animation)
- void createSpecialAttackCooldownTimer()
- void updateBroadSwordSpecialAttack(States,States,Animation)
- void jumpBroadSword(Animation)
- void createBroadSwordLastAttack(States,Animation)
- void updateDaggerSpecialAttack(States,States,Animation)
- void createCritRateBuffTimer()
MonsterAnimationThread
- ArrayList<States> bossAnimationState
- int frameCount
- int speedMultipiler
- boolean isBossJump
+ «Create» MonsterAnimationThread(BaseScene)
+ void run()
- void makeBossFrameAnimation()
- void updateBossAnimationFrame(States,States,BaseBoss,Animation)
- void createBossCooldown(BaseBoss,States)
- void updateBossSkillOne(States,States,BaseBoss,Animation)
- void createStompEffect(BaseBoss,Animation)
- void updateBossSkillTwoFirstPart(States,States,BaseBoss,Animation)
- void jumpBossToPlayer(BaseBoss,Animation)
- void updateBossSkillTwoLastPart(States,BaseBoss,Animation)
- void createStompEffectAndShockWave(BaseBoss,Animation)
- void createMultipleShockWave(BaseBoss,Animation)
- void createShockWave(BaseBoss,Animation,FaceDirection,double,double,int)
- void updateBossSkillThree(States,States,BaseBoss,Animation)
- void createFire(BaseBoss,Animation)

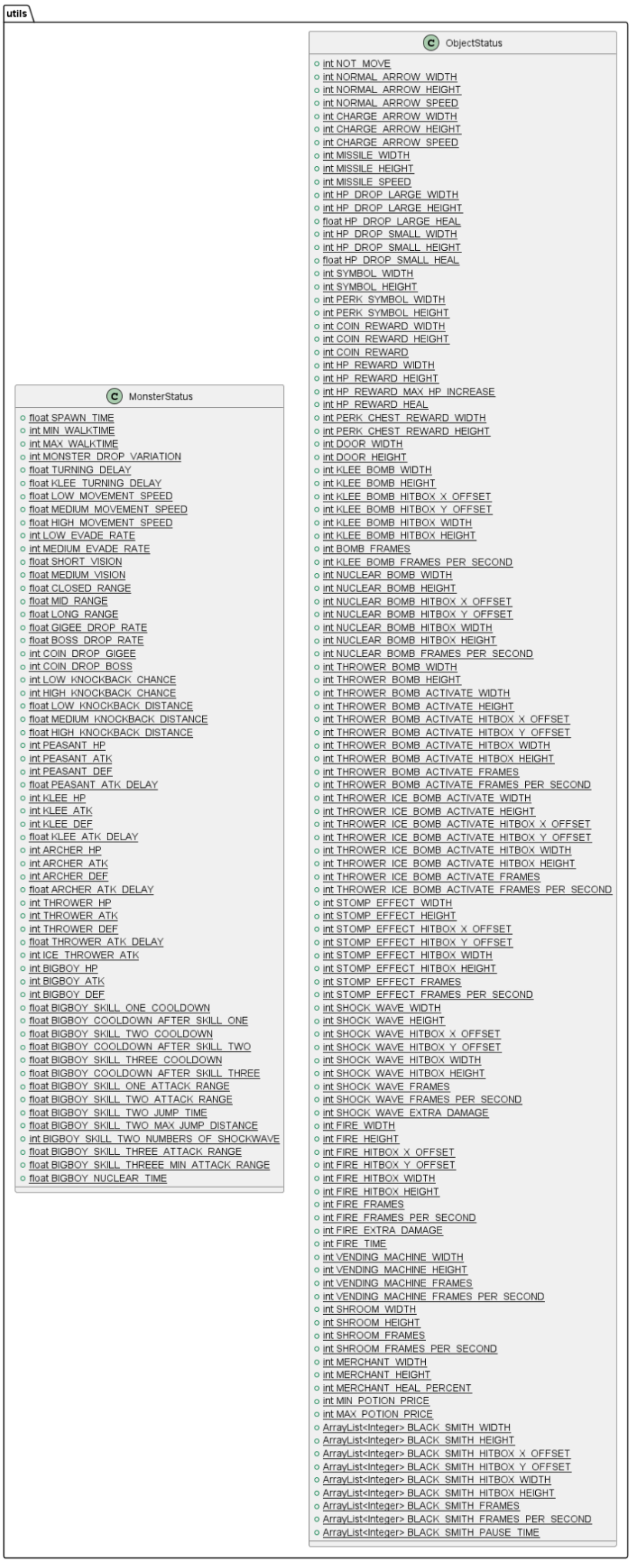
4.17. Utils


utils
HelperMethods
Random random
SoundLoader PLAYER_WALK
SoundLoader PLAYER_DASH
SoundLoader PLAYER_JUMP1
SoundLoader PLAYER_JUMP2
SoundLoader PLAYER_COLLECT
SoundLoader PLAYER_UPGRADE
SoundLoader PLAYER_BEING_HIT
SoundLoader PLAYER_DODGE
SoundLoader PLAYER_BUY_SUCCESS
SoundLoader PLAYER_BUY_FAIL
SoundLoader PLAYER_BUY_MISS
SoundLoader PLAYER_BRUH
SoundLoader PLAYER_DEAD
SoundLoader MONSTER_BEING_HIT
SoundLoader MONSTER_DEAD
SoundLoader MONSTER_DODGE
SoundLoader BOSS_FIRE
SoundLoader BOSS_STOMP_EFFECT
SoundLoader DAGGER_COMBO_ATTACK1
SoundLoader DAGGER_COMBO_ATTACK2
SoundLoader BROADSWORD_ATTACK
SoundLoader BROADSWORD_STOMP_EFFECT
SoundLoader NORMAL_ARROW_ATTACK
SoundLoader CHARGE_ARROW_CHARGE
SoundLoader CHARGE_ARROW_ATTACK
SoundLoader MISSILE_ATTACK
SoundLoader THROWERBOMB_ATTACK
SoundLoader KLEE_BOMB
SoundLoader NUCLEAR_BOMB
SoundLoader GET_HEAL
SoundLoader MERCHANT_SELL
SoundLoader PERKCHEST_GET
SoundLoader BASEREWARD_GET
SoundLoader DOOR_OPEN
int randomInteger(int,int)
float randomFloat(float,float)
float randomChance()
int randomIntegerUsingPercent(int,int)
int finalValuePercent(int,int)
float finalValuePercentFloat(float,int)
int calculateActionAttack(BaseEntity,BaseEntity,boolean,int)
int calculateReflectionAttack(int,BaseMonster)
void removeFromUnobtain(Player,BasePerk)
LevelData
int NUMBER_OF_FIGHT_ROOM_BEFORE_MARKET
int[][] classSelectionSceneData
int[][] firstFightSceneData
int[][] secondFightSceneData
int[][] thirdFightSceneData
int[][] forthFightSceneData
int[][] fifthFightSceneData
int[][] marketSceneData
int[][] BossSceneData
ArrayList<int[][]> getAllFightScene()

- ArrayList<BasePerk> generateAllPerks(Player)
- Button createButton(String)
- Button createButton(String,Color,Color)
- Text createTextForInventory(String,int,Color,double,double)
- Text createText(String,int,Color)
- Text createText(String,int,Color,FontWeight)
- boolean canMoveHere(BaseEntity,double,double,double,double,int[][])
- boolean isSolid(BaseEntity,double,double,int[][])
- boolean canMoveHereForObject(double,double,double,double,int[][])
- boolean isSolidForObject(double,double,int[][])
- void spawnPlayer(Player,int[][])
- int getTypeOfTile(double,double,int[][])
- void spawnMonsters(int[][],BaseScene)
- void spawnMonster(Player,String,BaseScene,double,double)
- ArrayList<SoundLoader> loadEntitiesSound(BaseEntity)
- ArrayList<SoundLoader> loadObjectsSound(BaseObject)
- SoundLoader getSoundFromSoundList(ArrayList<SoundLoader>,String)
- String randomRoomType()
- void calculateNewHp(BaseEntity,int,int)
- int countMonstersNearby(Player,float,float)
- boolean hasPerk(Player,BasePerk,boolean)
- ArrayList<BasePerk> createPerkChoices(Player,ArrayList<BasePerk>,int)
- void addPerkToResult(BasePerk,ArrayList<BasePerk>)
- ArrayList<String> createDoorChoices(ArrayList<String>,int)
- void createColumns(GridPane,int[])
- void createRows(GridPane,int[])
- ArrayList<SoundLoader> getAllSounds()
- void addAndDeleteText(BaseEntity,BaseScene,Text)
- void addAndDeleteStageStatus(BaseScene,Text)
- void knockBack(BaseEntity,FaceDirection,BaseScene)
- void immuneAfterAttacked(BaseEntity)
- void resetAllScenes()
- void createSymbol(BaseScene,String,double,double,BasePerk)
- ImageView loadCoinUI()


javafx
scene
canvas
Canvas
utils
SoundLoader
String name
MediaPlayer mediaPlayer
AudioClip audioClip
ArrayList<String> BACKGROUNDS
boolean isPlay
void play()
void stop()
String getName()
MediaPlayer getMediaPlayer()
GameConstants
int gameWidth
int gameHeight
int uiOffset
int healthWidth
int healthHeight
int healthUIWidth
int healthUIHeight
double scaleForUI
double edgeRightOfHealthBar
double edgeTopOfHealthBar
ArrayList<String> ROOM_TYPES
double borderL
double borderR
SoundLoader START_SCENE_SONG
SoundLoader CLASS_SELECTION_SCENE_SONG
SoundLoader FIGHT_SCENE_SONG1
SoundLoader FIGHT_SCENE_SONG2
SoundLoader FIGHT_SCENE_SONG3
SoundLoader FIGHT_SCENE_SONG4
SoundLoader FIGHT_SCENE_SONG5
SoundLoader FIGHT_SCENE_SONG6
SoundLoader MARKET_SCENE_SONG
SoundLoader BOSS_SCENE_SONG
SoundLoader INVENTORY
ArrayList<Integer> passableTile
ArrayList<Integer> airTile
ArrayList<Integer> spawnMonsterTile
Font thaleahFatFont
Font thaleahFatFontButton
Color GREEN_FOR_UI
double soundVolume
void setGameWidth(int)
void setGameHeight(int)
void stopAllBackgroundSongs()
CanvasDrawer
Image template
BaseScene scene
int getValueInThatCoor(int,int)
int getXCoor(int,int)
int getYCoor(int,int)
void drawBackground(GraphicsContext)
void drawDoors(ArrayList<String>,int,int)
void drawMinimap(GraphicsContext)
void drawEntities(GraphicsContext)
void drawDoorsMinimap(GraphicsContext)
void drawPlayerMinimap(GraphicsContext)
void drawAllMonstersMinimap(GraphicsContext)


utils
PerkStatus
float PRICE_UPGRADE_PERCENT
int PRICE_VARIATION
int COMMON_PRICE
int UNCOMMON_PRICE
int RARE_PRICE
float BERSERK_1_CRITERIA
float BERSERK_2_1_CRITERIA
float BERSERK_2_2_CRITERIA
int BERSERK_1_ADDATK
int BERSERK_2_1_ADDATK
int BERSERK_2_2_ADDATK
float DISCOUNT_NORMAL_FAILCHANCE
float DISCOUNT_2_2_FAILCHANCE
int DISCOUNT_1_DISCOUNT
int DISCOUNT_2_1_DISCOUNT
int DISCOUNT_2_2_DISCOUNT
float EXTROVERT_TILES_X
float EXTROVERT_TILES_Y
int EXTROVERT_CRITERIA
int EXTROVERT_1_ADDATKPERMONSTER
int EXTROVERT_2_1_ADDATKPERMONSTER
int EXTROVERT_2_2_ADDATKPERMONSTER
int EXTROVERT_2_2_MINUSATKEXTRA
int FATAL_1_ADDCRITDAMAGE
int FATAL_2_1_ADDCRITDAMAGE
int FATAL_2_2_ADDCRITDAMAGE
int FATAL_2_2_MINUSCRITRATE
int FORTIFY_1_ADDDAMAGEDECREASE
int FORTIFY_2_1_ADDDAMAGEDECREASE
int FORTIFY_2_2_ADDDAMAGEDECREASE
int FORTIFY2_2_ADDATKDELAY
float INTROVERT_TILES_X
float INTROVERT_TILES_Y
int INTROVERT_1_BASEADDATK
int INTROVERT_1_MINUSADDATKPERMONSTER
int INTROVERT_2_1_BASEADDATK
int INTROVERT_2_1_MINUSADDATKPERMONSTER
int INTROVERT_2_2_BASEADDATK
int INTROVERT_2_2_MINUSADDATKPERMONSTER
int JUKE_1_ADDEVADERATE
int JUKE_2_1_ADDEVADERATE
int JUKE_2_2_ADDEVADERATE
int JUKE_2_2_MINUSDAMAGEDECREASE
float LUCKYMAN_NORMAL_TIERCHANCE
float LUCKYMAN_2_1_TIERCHANCE
float LUCKYMAN_2_2_DROPCHANCE
float PERFECTION_1_CRITERIA
float PERFECTION_2_1_CRITERIA
float PERFECTION_2_2_CRITERIA
int PERFECTION_1_ADDATK
int PERFECTION_2_1_ADDATK
int PERFECTION_2_2_ADDATK
int PRECISION_1_ADDCRITRATE
int PRECISION_2_1_ADDCRITRATE
int PRECISION_2_2_ADDCRITRATE
int PRECISION_2_2_MINUSCRITDAMAGE
int RAPID_NORMAL_ADDATKSPEED
int RAPID_2_1_ADDATKSPEED
int RAPID_2_2_ADDMOVEMENTSPEED
int REINFORCED_1_ADDDEF
int REINFORCED_2_1_ADDDEF
int REINFORCED_2_2_ADDDEF
int REINFORCED_2_2_MINUSMOVEMENTSPEED
float THORN_NORMAL_REFLECTCHANCE
float THORN_2_1_REFLECTCHANCE
int THORN_2_2_SLOWTIME
int THORN_2_2_SLOW
float TREASURE_NORMAL_REWARDCHANCE
float TREASURE_2_1_REWARDCHANCE
float TREASURE_2_2_REWARDMULTIPLIER
int UNDEAD_NORMAL_ACTIVATION
int UNDEAD_2_1_ACTIVATION
int UNDEAD_NORMAL_COOLDOWN
int UNDEAD_2_2_COOLDOWN
int UNDEAD_2_1_ADDATK
int UNDEAD_2_1_ADDATKSPEED
float VAMP_NORMAL_DRAINCHANCE
float VAMP_2_2_DRAINCHANCE
float VAMP_NORMAL_DRAINPERHIT
float VAMP_2_1_DRAINPERHIT
PlayerStatus
int START_COINS
int MAX_PERK_OBTAIN
int LOW_HP
int HIGH_HP
int LOW_ATK
int HIGH_ATK
int LOW_DEF
int HIGH_DEF
float LOW_MOVEMENT_SPEED
float MEDIUM_MOVEMENT_SPEED
float HIGH_MOVEMENT_SPEED
float DAGGER_NORMAL_ATTACK_SPEED_MULTIPLIER
float DAGGER_SPECIAL_ATTACK_SPEED_MULTIPLIER
float BOW_NORMAL_ATTACK_SPEED_MULTIPLIER
float BOW_SPECIAL_ATTACK_SPEED_MULTIPLIER
float BROADSWORD_NORMAL_ATTACK_SPEED_MULTIPLIER
float BROADSWORD_SPECIAL_ATTACK_SPEED_MULTIPLIER
int LOW_EVADE_RATE
int HIGH_EVADE_RATE
int LOW_CRIT_RATE
int HIGH_CRIT_RATE
int LOW_CRIT_DAMAGE
int HIGH_CRIT_DAMAGE
int NORMAL_MAX_DASH
int LOW_MAX_DASH
float DEFAULT_DASH_DISTANCE
int DEFAULT_DASH_TIME
float DASH_COOLDOWN
float TIME_CAN_SECOND_ATTACK
float PLAYER_DAMAGE_COOLDOWN


utils
MonsterStatus
float SPAWN_TIME
int MIN_WALKTIME
int MAX_WALKTIME
int MONSTER_DROP_VARIATION
float TURNING_DELAY
float KLEE_TURNING_DELAY
float LOW_MOVEMENT_SPEED
float MEDIUM_MOVEMENT_SPEED
float HIGH_MOVEMENT_SPEED
int LOW_EVADE_RATE
int MEDIUM_EVADE_RATE
float SHORT_VISION
float MEDIUM_VISION
float CLOSED_RANGE
float MID_RANGE
float LONG_RANGE
float GIGEE_DROP_RATE
float BOSS_DROP_RATE
int COIN_DROP_GIGEE
int COIN_DROP_BOSS
int LOW_KNOCKBACK_CHANCE
int HIGH_KNOCKBACK_CHANCE
float LOW_KNOCKBACK_DISTANCE
float MEDIUM_KNOCKBACK_DISTANCE
float HIGH_KNOCKBACK_DISTANCE
int PEASANT_HP
int PEASANT_ATK
int PEASANT_DEF
float PEASANT_ATK_DELAY
int KLEE_HP
int KLEE_ATK
int KLEE_DEF
float KLEE_ATK_DELAY
int ARCHER_HP
int ARCHER_ATK
int ARCHER_DEF
float ARCHER_ATK_DELAY
int THROWER_HP
int THROWER_ATK
int THROWER_DEF
float THROWER_ATK_DELAY
int ICE_THROWER_ATK
int BIGBOY_HP
int BIGBOY_ATK
int BIGBOY_DEF
float BIGBOY_SKILL_ONE_COOLDOWN
float BIGBOY_COOLDOWN_AFTER_SKILL_ONE
float BIGBOY_SKILL_TWO_COOLDOWN
float BIGBOY_COOLDOWN_AFTER_SKILL_TWO
float BIGBOY_SKILL_THREE_COOLDOWN
float BIGBOY_COOLDOWN_AFTER_SKILL_THREE
float BIGBOY_SKILL_ONE_ATTACK_RANGE
float BIGBOY_SKILL_TWO_ATTACK_RANGE
float BIGBOY_SKILL_TWO_JUMP_TIME
float BIGBOY_SKILL_TWO_MAX_JUMP_DISTANCE
int BIGBOY_SKILL_TWO_NUMBERS_OF_SHOCKWAVE
float BIGBOY_SKILL_THREE_ATTACK_RANGE
float BIGBOY_SKILL_THREEE_MIN_ATTACK_RANGE
float BIGBOY_NUCLEAR_TIME
ObjectStatus
int NOT_MOVE
int NORMAL_ARROW_WIDTH
int NORMAL_ARROW_HEIGHT
int NORMAL_ARROW_SPEED
int CHARGE_ARROW_WIDTH
int CHARGE_ARROW_HEIGHT
int CHARGE_ARROW_SPEED
int MISSILE_WIDTH
int MISSILE_HEIGHT
int MISSILE_SPEED
int HP_DROP_LARGE_WIDTH
int HP_DROP_LARGE_HEIGHT
float HP_DROP_LARGE_HEAL
int HP_DROP_SMALL_WIDTH
int HP_DROP_SMALL_HEIGHT
float HP_DROP_SMALL_HEAL
int SYMBOL_WIDTH
int SYMBOL_HEIGHT
int PERK_SYMBOL_WIDTH
int PERK_SYMBOL_HEIGHT
int COIN_REWARD_WIDTH
int COIN_REWARD_HEIGHT
int COIN_REWARD
int HP_REWARD_WIDTH
int HP_REWARD_HEIGHT
int HP_REWARD_MAX_HP_INCREASE
int HP_REWARD_HEAL
int PERK_CHEST_REWARD_WIDTH
int PERK_CHEST_REWARD_HEIGHT
int DOOR_WIDTH
int DOOR_HEIGHT
int KLEE_BOMB_WIDTH
int KLEE_BOMB_HEIGHT
int KLEE_BOMB_HITBOX_X_OFFSET
int KLEE_BOMB_HITBOX_Y_OFFSET
int KLEE_BOMB_HITBOX_WIDTH
int KLEE_BOMB_HITBOX_HEIGHT
int BOMB_FRAMES
int KLEE_BOMB_FRAMES_PER_SECOND
int NUCLEAR_BOMB_WIDTH
int NUCLEAR_BOMB_HEIGHT
int NUCLEAR_BOMB_HITBOX_X_OFFSET
int NUCLEAR_BOMB_HITBOX_Y_OFFSET
int NUCLEAR_BOMB_HITBOX_WIDTH
int NUCLEAR_BOMB_HITBOX_HEIGHT
int NUCLEAR_BOMB_FRAMES_PER_SECOND
int THROWER_BOMB_WIDTH
int THROWER_BOMB_HEIGHT
int THROWER_BOMB_ACTIVATE_WIDTH
int THROWER_BOMB_ACTIVATE_HEIGHT
int THROWER_BOMB_ACTIVATE_HITBOX_X_OFFSET
int THROWER_BOMB_ACTIVATE_HITBOX_Y_OFFSET
int THROWER_BOMB_ACTIVATE_HITBOX_WIDTH
int THROWER_BOMB_ACTIVATE_HITBOX_HEIGHT
int THROWER_BOMB_ACTIVATE_FRAMES
int THROWER_BOMB_ACTIVATE_FRAMES_PER_SECOND
int THROWER_ICE_BOMB_ACTIVATE_WIDTH
int THROWER_ICE_BOMB_ACTIVATE_HEIGHT
int THROWER_ICE_BOMB_ACTIVATE_HITBOX_X_OFFSET
int THROWER_ICE_BOMB_ACTIVATE_HITBOX_Y_OFFSET
int THROWER_ICE_BOMB_ACTIVATE_HITBOX_WIDTH
int THROWER_ICE_BOMB_ACTIVATE_HITBOX_HEIGHT
int THROWER_ICE_BOMB_ACTIVATE_FRAMES
int THROWER_ICE_BOMB_ACTIVATE_FRAMES_PER_SECOND
int STOMP_EFFECT_WIDTH
int STOMP_EFFECT_HEIGHT
int STOMP_EFFECT_HITBOX_X_OFFSET
int STOMP_EFFECT_HITBOX_Y_OFFSET
int STOMP_EFFECT_HITBOX_WIDTH
int STOMP_EFFECT_HITBOX_HEIGHT
int STOMP_EFFECT_FRAMES
int STOMP_EFFECT_FRAMES_PER_SECOND
int SHOCK_WAVE_WIDTH
int SHOCK_WAVE_HEIGHT
int SHOCK_WAVE_HITBOX_X_OFFSET
int SHOCK_WAVE_HITBOX_Y_OFFSET
int SHOCK_WAVE_HITBOX_WIDTH
int SHOCK_WAVE_HITBOX_HEIGHT
int SHOCK_WAVE_FRAMES
int SHOCK_WAVE_FRAMES_PER_SECOND
int SHOCK_WAVE_EXTRA_DAMAGE
int FIRE_WIDTH
int FIRE_HEIGHT
int FIRE_HITBOX_X_OFFSET
int FIRE_HITBOX_Y_OFFSET
int FIRE_HITBOX_WIDTH
int FIRE_HITBOX_HEIGHT
int FIRE_FRAMES
int FIRE_FRAMES_PER_SECOND
int FIRE_EXTRA_DAMAGE
int FIRE_TIME
int VENDING_MACHINE_WIDTH
int VENDING_MACHINE_HEIGHT
int VENDING_MACHINE_FRAMES
int VENDING_MACHINE_FRAMES_PER_SECOND
int SHROOM_WIDTH
int SHROOM_HEIGHT
int SHROOM_FRAMES
int SHROOM_FRAMES_PER_SECOND
int MERCHANT_WIDTH
int MERCHANT_HEIGHT
int MERCHANT_HEAL_PERCENT
int MIN_POTION_PRICE
int MAX_POTION_PRICE
ArrayList<Integer> BLACK_SMITH_WIDTH
ArrayList<Integer> BLACK_SMITH_HEIGHT
ArrayList<Integer> BLACK_SMITH_HITBOX_X_OFFSET
ArrayList<Integer> BLACK_SMITH_HITBOX_Y_OFFSET
ArrayList<Integer> BLACK_SMITH_HITBOX_WIDTH
ArrayList<Integer> BLACK_SMITH_HITBOX_HEIGHT
ArrayList<Integer> BLACK_SMITH_FRAMES
ArrayList<Integer> BLACK_SMITH_FRAMES_PER_SECOND
ArrayList<Integer> BLACK_SMITH_PAUSE_TIME


utils
AnimationStatus
double SPAWN_X
double SPAWN_Y
int START_X
int START_Y
int DEFAULT_ANIMATION_WIDTH
int DEFAULT_ANIMATION_HEIGHT
int GAP_WIDTH
int GAP_HEIGHT
int DEFAULT_HITBOX_WIDTH
int DEFAULT_HITBOX_HEIGHT
int DEFAULT_HITBOX_X_OFFSET
int DEFAULT_HITBOX_Y_OFFSET
float SCALE
int PIXEL_PER_TILE
int MOVE_FRAME
int ANIMATION_FRAME
float MOVE_PER_FRAME
float JUMP_HEIGHT
float GRAVITY
float GRAVITY_FOR_OBJECT
double JUMP_VELOCITY
int JUMP_FRAME
ArrayList<String> DASH_AND_JUMP_WHEN_DEATH_MONSTER
int ANONYMOUS_STAND_STILL_FRAMES
int ANONYMOUS_WALK_FRAMES
int BOW_STAND_STILL_FRAMES
int BOW_WALK_FRAMES
int BOW_NORMAL_ATTACK_FRAMES
int BOW_SPECIAL_ATTACK_FRAMES
int BOW_DASH_FRAME
int BOW_DEAD_FRAMES
int BOW_SPECIAL_ATTACK_CHARGE_TIME
float BOW_SPECIAL_ATTACK_COOLDOWN
int BOW_SPECIAL_ATTACK_DAMAGE
int BOW_SPECIAL_ATTACK_DAMAGE_INCREASE_PER_FRAME
int BOW_SPECIAL_ATTACK_MAX_DAMAGE_INCREASE
int BROADSWORD_STAND_STILL_FRAMES
int BROADSWORD_WALK_FRAMES
int BROADSWORD_NORMAL_ATTACK_FRAMES
int BROADSWORD_SPECIAL_ATTACK_FRAMES
int BROADSWORD_DASH_FRAMES
int BROADSWORD_DEAD_FRAMES
int BROADSWORD_ANIMATION_WIDTH
int BROADSWORD_ANIMATION_HEIGHT
int BROADSWORD_HITBOX_WIDTH
int BROADSWORD_HITBOX_HEIGHT
int BROADSWORD_HITBOX_X_OFFSET
int BROADSWORD_HITBOX_Y_OFFSET
int BROADSWORD_SPECIAL_ATTACK_JUMP_HEIGHT
float BROADSWORD_SPECIAL_ATTACK_COOLDOWN
int BROADSWORD_SPECIAL_ATTACK_DAMAGE
int DAGGER_STAND_STILL_FRAMES
int DAGGER_WALK_FRAMES
int DAGGER_NORMAL_ATTACK_FRAMES
int DAGGER_SPECIAL_ATTACK_FRAMES
int DAGGER_DASH_FRAMES
int DAGGER_DEAD_FRAMES
float DAGGER_SPECIAL_ATTACK_WARP_DISTANCE
float DAGGER_SPECIAL_ATTACK_COOLDOWN
int DAGGER_SPECIAL_ATTACK_DAMAGE
int DAGGER_SPECIAL_ATTACK_CRIT_CHANCE
float DAGGER_SPECIAL_ATTACK_CRIT_BUFF_DURATION
int SPAWN_FRAMES
int PEASANTGIGEE_STAND_STILL_FRAMES
int PEASANTGIGEE_WALK_FRAMES
int PEASANTGIGEE_NORMAL_ATTACK_FRAMES
int PEASANTGIGEE_DEAD_FRAMES
int KLEEGIGEE_STAND_STILL_FRAMES
int KLEEGIGEE_WALK_FRAMES
int KLEEGIGEE_NORMAL_ATTACK_FRAMES
int KLEEGIGEE_DEAD_FRAMES
int KLEEGIGEE_SPEED_INCREASE
float KLEEGIGEE_BOMB_TIME
int ARCHERGIGEE_STAND_STILL_FRAMES
int ARCHERGIGEE_WALK_FRAMES
int ARCHERGIGEE_NORMAL_ATTACK_FRAMES
int ARCHERGIGEE_DEAD_FRAMES
int THROWERGIGEE_STAND_STILL_FRAMES
int THROWERGIGEE_WALK_FRAMES
int THROWERGIGEE_NORMAL_ATTACK_FRAMES
int THROWERGIGEE_DEAD_FRAMES
int THROWERGIGEE_HITBOX_WIDTH
int THROWERGIGEE_HITBOX_X_OFFSET
float THROWERGIGEE_THROW_TIME
int ICETHROWERGIGEE_STAND_STILL_FRAMES
int ICETHROWERGIGEE_WALK_FRAMES
int ICETHROWERGIGEE_NORMAL_ATTACK_FRAMES
int ICETHROWERGIGEE_DEAD_FRAMES
float ICETHROWERGIGEE_THROW_TIME
int ICETHROWERGIGEE_SLOW_AMOUNT
int ICETHROWERGIGEE_MAX_SLOW_AMOUNT
int ICETHROWERGIGEE_SLOW_DURATION
int BIGBOY_STAND_STILL_FRAMES
int BIGBOY_WALK_FRAMES
int BIGBOY_SKILL_ONE_FRAMES
ArrayList<Integer> BIGBOY_SKILL_ONE_FRAMES_HOLD
float BIGBOY_SKILL_ONE_ANIMATION_SPEED
int BIGBOY_SKILL_TWO_FRAMES
ArrayList<Integer> BIGBOY_SKILL_TWO_FRAMES_HOLD
float BIGBOY_SKILL_TWO_ANIMATION_SPEED
int BIGBOY_SKILL_THREE_STAND_FRAMES
int BIGBOY_SKILL_THREE_WALK_FRAMES
int BIGBOY_DEAD_FRAMES
int BIGBOY_ANIMATION_WIDTH
int BIGBOY_ANIMATION_HEIGHT
int BIGBOY_HITBOX_WIDTH
int BIGBOY_HITBOX_HEIGHT
int BIGBOY_HITBOX_X_OFFSET
int BIGBOY_HITBOX_Y_OFFSET

5. Packages

5.1. Package animating

5.1.1. Class Animation extends ImageView

- This class is used for controling entities' animation and objects' animation on the screen

Fields

Name	Description
- BaseEntity baseEntity	This animation belongs to this baseEntity
- ColorAdjust colorAdjust (= new ColorAdjust())	For adjusting the animation's brightness
- int currentFrameIndex	Current Index in the animation sprite
- States previousState	Latest state of the animation
- States state	Current state of the animation
- double yVelocity (= 0)	Speed in y-axis
- boolean isMoveLeft (= false)	Boolean for checking whether baseEntity is moving left or not
- boolean isMoveRight (= false)	Boolean for checking whether baseEntity is moving right or not
- boolean willMoveLeft (= false)	Boolean for checking whether baseEntity will move left or not
- boolean willMoveRight (= false)	Boolean for checking whether baseEntity will move right or not
- boolean isAnimationPlaying (= false)	Boolean for checking whether the animation is running or not
- boolean isHoldRMB (= false)	Boolean for checking whether player press RMB key or not
- boolean isJump (= false)	Boolean for checking whether baseEntity is jumping or not
- boolean canJump (= true)	Boolean for checking whether baseEntity can jump or not

Constructors

Name	Description

+ Animation(BaseEntity baseEntity, double posX, double posY)	Constructor used in BaseEntity's Animation
	- Set image to baseEntity's image
	- Set baseEntity to baseEntity
	- Set fit height to baseEntity's animation width
	- Set fit width to baseEntity's animation height
	- If baseEntity instanceof Player
	  - Set state to PLAYER_STAND_STILL
	  - Set previous state to PLAYER_STAND_STILL
	  - Update frame with state PLAYER_STAND_STILL
	- If baseEntity instanceof BaseBoss
	  - Set state to MONSTER_STAND_STILL
	  - Set previous state to MONSTER_STAND_STILL
	  - Update frame with state MONSTER_STAND_STILL
	- If baseEntity instanceof BaseMonster
	  - Set state to MONSTER_SPAWN
	  - Set previous state to MONSTER_SPAWN
	  - Update frame with state MONSTER_SPAWN
	- Set spawn point to posX, posY and set current frame index to 0

+ Animation(BaseEntity baseObject, double posX, double posY)	Constructor used in BaseObject's Animation - Call super() with object's image string - Set state to OBJECT - Set fit height to object's height - Set fit width to object's width - Set spawn point to posX, posY

Methods

Name	Description
+ void setSpawnPoint(BaseEntity entity, double posX, double posY)	Set spawn point of entity - If entity's face direction is left, set scaleX to -1 - Set translateX, translateY at posX, posY - Set hitBox translateX, translateY
+ void setSpawnPoint(BaseObject object, double posX, double posY)	Set spawn point of object - If object's face direction is left, set scaleX to -1 - Set translateX, translateY at posX, posY
+ void updateFrame(States state)	Update animation frame - Check column and row of state - Find position of animation - Set viewport to that position and increase current frame index by 1
+ int getNumberOfRows(BaseEntity baseEntity, States state)	Check number of rows for the entity in each state - Return number of rows
+ ColorAdjust getDefaultBrightness()	- Set brightness to 0 - Return colorAdjust
+ void setAnimationToDefault(BaseEntity entity, BaseScene scene)	Set animation to default animation - If entity is player - If player press movement key - Set state to PLAYER_MOVE - Set scaleX and moveLeft, moveRight to match the movement key - Set current frame index to 0 - If not, set state to PLAYER_STAND_STILL and set moveLeft, moveRight to false. - If entity is monster - Set state to MONSTER_STAND_STILL

	- Set current frame index to 0
+ void setDisableMovement()	Disable all movement - Set state to PLAYER_STAND_STILL - Set current frame index to 0 - Set moveLeft, moveRight, willMoveLeft, willMoveRight to false
+ BaseEntity getBaseEntity()	Getter for baseEntity
+ void setBaseEntity(BaseEntity baseEntity)	Setter for baseEntity
+ int getCurrentFrameIndex()	Getter for currentFrameIndex
+ void setCurrentFrameIndex(int currentFrameIndex)	Setter for currentFrameIndex
+ States getState()	Getter for state
+ void setState(States state)	Setter for state
+ boolean isMoveLeft()	Getter for isMoveLeft
+ void setMoveLeft(boolean moveLeft)	Setter for isMoveLeft - Set baseEntity' face direction to LEFT
+ boolean isMoveRight()	Getter for isMoveRight
+ void setMoveRight(boolean moveRight)	Setter for isMoveRight - Set baseEntity's face direction to RIGHT
+ boolean willMoveLeft()	Getter for willMoveLeft
+ void setWillMoveLeft(boolean willMoveLeft)	Setter for willMoveLeft
+ boolean willMoveRight()	Getter for willMoveRight
+ void setWillMoveRight(boolean willMoveRight)	Setter for willMoveRight
+ boolean canJump()	Getter for canJump
+ void setCanJump(boolean canJump)	Setter for canJump
+ boolean isAnimationPlaying()	Getter for isAnimationPlaying

+ void setAnimationPlaying(boolean animationPlaying)	Setter for isAnimationPlaying
+ boolean isHoldRMB()	Getter for isHoldRMB
+ void setHoldRMB(boolean holdRMB)	Setter for isHoldRMB
+ States getPreviousState()	Getter for previousState
+ void setPreviousState(States previousState)	Setter for previousState
+ double getyVelocity()	Getter for yVelocity
+ void setyVelocity(double yVelocity)	Setter for yVelocity

5.1.2. Class HitBox extends ImageView

- This class is used for describing the hitbox of entities and objects

Fields

Name	Description
- BaseEntity baseEntity	This hitbox belongs to this baseEntity

Constructors

Name	Description
+ HitBox(BaseEntity baseEntity)	Constructor for BaseEntity - Set baseEntity to baseEntity - Set position based on animation and entity's hitbox offset - Set fit width to entity's hitbox width - Set fit height to entity's hitbox height
+ HitBox(BaseObject baseObject)	Constructor for BaseObject using only BaseObject - Call HitBox() with object's width, object's height, 0, 0
+ HitBox(BaseObject baseObject, int width, int height, int xOffset, int yOffset)	Constructor for BaseObject - Set position based on animation and object's hitbox offset - Set fit width to object's hitbox

	width - Set fit height to objet's hitbox height

Methods

Name	Description
+ BaseEntity getBaseEntity()	Getter for baseEntity
+ void setBaseEntity(BaseEntity baseEntity)	Setter for baseEntity

5.2. Package application

5.2.1. Class Main extends Application

- This class is used to launch the program

Method

Name	Description
+ void main(String[] args)	The entry point of the application
+ void start(Stage primaryStage) throws Exception	Launching the StartScene

5.3. Package entities.bases

5.3.1. Abstract Class BaseBoss extends BaseMonster

- This class is used as a base class of the BigBoy class

Fields

Name	Description
- float cooldownSkill1	Cooldown for skill 1
- float cooldownSkill2	Cooldown for skill 2
- float cooldownSkill3	Cooldown for skill 3
- float skillOneAttackRange	Attack range for skill 1
- float skillTwoAttackRange	Attack range for skill 2

- float skillThreeAttackRange	Attack range for skill 3
- boolean canSkill1 (= true)	Boolean for checking whether that can use skill 1
- boolean canSkill2 (= true)	Boolean for checking whether that can use skill 2
- boolean canSkill3 (= true)	Boolean for checking whether that can use skill 3
- ArrayList<Integer> skillToRandom (= new ArrayList<>())	Arraylist for contain skill that can be used
- float attackDelay (= 0)	Delay time after using skill
- ArrayList<Integer> frameMakeDamageSkillOne (= new ArrayList<>())	Arraylist for contain frame that can make damage in skill 1
- ArrayList<Integer> frameMakeDamageSkillTwo (= new ArrayList<>())	Arraylist for contain frame that can make damage in skill 2
- ArrayList<Integer> frameMakeDamageSkillThree (= new ArrayList<>())	Arraylist for contain frame that can make damage in skill 3

Constructors

Name	Description
+ BaseBoss()	Constructor for BaseBoss - super() - Set common status

Methods

Name	Description
+ void setCommonStatus()	Set status to boss - Set evade rate to MEDIUM_EVADE_RATE - Set drop rate to BOSS_DROP_RATE - Set coin drop to COIN_DROP_RATE - Set knockback chance to LOW_KNOCKBACK_CHANCE - Set knockback distance to

	LOW_KNOCKBACK_DiSTANCE
+ float getCooldownSkill1()	Getter for cooldownSkill1
+ void setCooldownSkill1(float coolDownSkill1)	Setter for cooldownSkill1
+ float getCooldownSkill2()	Getter for cooldownSkill2
+ void setCooldownSkill2(float coolDownSkill2)	Setter for cooldownSkill2
+ float getCooldownSkill3()	Getter for cooldownSkill3
+ void setCooldownSkill3(float coolDownSkill3)	Setter for cooldownSkill3
+ float getSkillOneAttackRange()	Getter for skillOneAttackRange
+ void setSkillOneAttackRange(float skillOneAttackRange)	Setter for skillOneAttackRange
+ float getSkillTwoAttackRange()	Getter for skillTwoAttackRange
+ void setSkillTwoAttackRange(float skillTwoAttackRange)	Setter for skillTwoAttackRange
+ float getSkillThreeAttackRange()	Getter for skillThreeAttackRange
+ void setSkillThreeAttackRange(float skillThreeAttackRange)	Setter for skillThreeAttackRange
+ ArrayList<Integer> getSkillToRandom()	Getter for skillToRandom
+ boolean canSkill1()	Getter for canSkill1
+ void setCanSkill1(boolean canSkill1)	Setter for canSkill1
+ boolean canSkill2()	Getter for canSkill2
+ void setCanSkill2(boolean canSkill2)	Setter for canSkill2
+ boolean canSkill3()	Getter for canSkil3
+ void setCanSkill3(boolean canSkill1)	Setter for canSkill3

+ float getAttackDelay()	Getter for attackDelay
+ void setAttackDelay(float attackDelay)	Setter for attackDelay
+ ArrayList<Integer> getFrameMakeDamageSkillOne( )	Getter for frameMakeDamageSkillOne
+ ArrayList<Integer> getFrameMakeDamageSkillTwo()	Getter for frameMakeDamgeSkillTwo
+ ArrayList<Integer> getFrameMakeDamageSkillThre e()	Getter for frameMakeDamageSkillThree

5.3.2. Abstract Class BaseClass

- This class is used as a base class of the player classes

Fields

Name	Description
- int hp	Class's health
- int atk	Class's attack
- int def	Class's defense
- float movementSpeed	Class's movement speed
- int evadeRate	Class's evade rate
- int damageIncrease	Class's damage increase
- int damageDecrease	Class's damage decrease
- FaceDirection faceDirection	Class's face direction
- float attackRange	Class's attack range
- float attackSpeed	Class's attack speed
- int critRate	Class's critical rate
- int critDamage	Class's critical damage
- String imgString	String for animation sprite image
- String uiString	String for UI image
- String inventoryUIString	String for inventory UI image

- int standStillFrames	Number of stand still frames
- int walkFrames	Number of walk frames
- int normalAttackFrames	Number of attack frames
- int specialAttackFrames	Number of special attack frames
- float specialAttackCooldown	Special attack cooldown time
- int dashFrames	Number of dash frames
- int deadFrames	Number of dead frames
- int attackCombo	Number of attack combo
- float timeCanSecondAttack	Time duration that still can second attack
- ArrayList<Integer> frameMakeDamageNormalAttack (= new ArrayList<>())	Arraylist for contain frame that can make damage in normal attack
- ArrayList<Integer> frameMakeDamageSpecialAttack (= new ArrayList<>())	Arraylist for contain frame that can make damage in special attack
- float knockbackDistance	Class's knockback distance
- int maxDash	Number of max dash
- int animationWidth	Animation width
- int animationHeight	Animation height
- int hitboxWidth	Hitbox width
- int hitboxHeight	Hitbox height
- int hitboxXOffset	Hitbox offset in x axis
- int hitboxYOffset	Hitbox offset in y axis

Constructors

Name	Description
+ BaseClass()	Constructor for BaseClass - Set common status

Methods

Name	Description
+ void setCommonStatus()	Set status to class - Set hp to LOW_HP - Set atk to LOW_ATK - Set def to LOW_DEF - Set movement speed to LOW_MOVEMENT_SPEED - Set evade rate to LOW_EVADE_RATE - Set damage increase to 0 - Set damage decrease to 0 - Set face direction to RIGHT - Set attack range to CLOST_RANGE - Set critical rate to LOW_CRIT_RATE - Set critical damage to LOW_CRITICAL_DAMAGE - Set max dash to NORMAL_MAX_DASH - Set animation width to DEFAULT_ANIMATIO_WIDTH - Set animation height to DEFAULT_ANIMATION_HEIGHT - Set hitbox width to DEFAULT_HITBOX_WIDTH - Set hitbox height to DEFAULE_HITBOX_HEIGHT - Set hitbox offset in axis X to DEFAULT_HITBOX_X_OFFSET - Set hitbox offset in axis Y to DEFAULT_HITBOX_Y_OFFSET - Set UI string - Set inventory UI string
+ *void setUniqueStatus()*	Set unique status
+ int getHp()	Getter for hp
+ void setHp(int hp)	Setter for hp - Hp can't below 0
+ int getAtk()	Getter for atk
+ void setAtk(int atk)	Setter for atk - Atk can’t below 1
+ int getDef()	Getter for def
+ void setDef(int def)	Setter for def

	- Def can’t below 0
+ float getMovementSpeed()	Getter for movementSpeed
+ void setMovementSpeed(float movementSpeed)	Setter for movement speed - Movement speed can’t below 0
+ int getEvadeRate()	Getter for evadeRate
+ void setEvadeRate(int evadeRate)	Setter for evadeRate - Evade rate can’t below 0
+ int getDamageIncrease()	Getter for damageIncrease
+ void setDamageIncrease(int damageIncrease)	Setter for damageIncrease
+ int getDamageDecrease()	Getter for damageDecrease
+ void setDamageDecrease(int damageDecrease)	Setter for damageDecrease
+ FaceDirection getFaceDirection()	Getter for faceDirection
+ void setFaceDirection(FaceDirection faceDirection)	Setter for faceDirection
+ float getAttackRange()	Getter for attackRange
+ void setAttackRange(float attackRange)	Setter for attackRange
+ float getAttackSpeed()	Getter for attackSpeed
+ void setAttackSpeesd(float attackSpeed)	Setter for attackSpeed - Attack speed can’t below 0
+ int getCritRate()	Getter for critRate
+ void setCritRate(int critRate)	Setter for critRate - Critical rate can’t below 0
+ int getCritDamage()	Getter for critDamage
+ void setCritDamage(int critDamage)	Setter for critDamage - Critical damage can’t below 0
+ String getImgString()	Getter for imageString
+ void setImgString(String imgString)	Setter for imageString

+ int getStandStillFrames()	Getter for standStillFrames
+ void setStandStillFrames(int standStillFrames)	Setter for standStillFrames
+ int getWalkFrames()	Getter for walkFrames
+ void setWalkFrames(int walkFrames)	Setter for walkFrames
+ int getNormalAttackFrames()	Getter for normalAttackFrames
+ void setNormalAttackFrames(int normalAttackFrames)	Setter for normalAttackFrames
+ int getSpecialAttackFrames()	Getter for specialAttackFrames
+ void setSpecialAttackFrames(int specialAttackFrames)	Setter for specialAttackFrames
+ int getDeadFrames()	Getter for deadFrames
+ void setDeadFrames(int deadFrames)	Setter for deadFrames
+ int getAttackCombo()	Getter for attackCombo
+ void setAttackCombo(int attackCombo)	Setter for attackCombo
+ float getTimeCanSecondAttack()	Getter for timeCanSecondAttack
+ void setTimeCanSecondAttack(float timeCanSecondAttack)	Setter for timeCanSecondAttack
+ ArrayList<Integer> getFrameMakeDamageNormalAttack()	Getter for frameMakeDamgeNormalAttack
+ ArrayList<Integer> getFrameMakeDamageSpecialAttack	Getter for frameMakeDamgeSpecialAttack
+ float getKnockbackDistance	Getter for knockbackDistance
+ void setKnockbackDistance(float knockbackDistance)	Setter for knockbackDistance
+ String getUiString()	Getter for uiString
+ void setUiString(String uiString)	Setter for uiString

+ String getInventoryUIString()	Getter for inventoryUIString
+ void setInventoryUIString(String inventoryUIString)	Setter for inventoryUIString
+ int getMaxDash()	Getter for maxDash
+ void setMaxDash(int maxDash)	Setter for maxDash
+ int getAnimationWidth()	Getter for animationWidth
+ void setAnimationWidth(int animationWidth)	Setter for animationWidth
+ int getAnimationHeight()	Getter for animationHeight
+ void setAnimationHeight(int animationHeight)	Setter for animationHeight
+ int getHitBoxWidth()	Getter for hitboxWidth
+ void setHitBoxWidth(int hitBoxWidth)	Setter for hitboxWidth
+ int getHitBoxHeight()	Getter for hitboxHeight
+ void setHitBoxHeight(int hitBoxHeight)	Setter for hitboxHeight
+ int getHitBoxXOffset()	Getter for hitBoxXOffset
+ void setHitBoxXOffset(int hitBoxXOffset)	Setter for hitBoxXOffset
+ int getHitBoxYOffset()	Getter for hitBoxYOffset
+ void setHitBoxYOffset(int hitBoxYOffset)	Setter for hitBoxYOffset
+ int getDashFrames()	Getter for dashFrames
+ void setDashFrames(int dashFrames)	Setter for dashFrames
+ float getSpecialAttackCoolDown()	Getter for specialAttackCooldown
+ void setSpecialAttackCoolDown(float specialAttackCoolDown)	Setter for specialAttackCooldown

5.3.3. Abstract Class BaseEntity implements Moveable, Attackable

- This class is used as a base class of entities in the game

Fields

Name	Description
- int ATTACK_VARIATION (= 5)	For create variation of damage
- int baseMaxHp	Base max health
- int addMaxHp	Additional max health
- int currentHp	Current health
- int baseAtk	Base attack
- int addAtk	Additional attack
- int baseDef	Base defense
- int addDef	Additional defense
- float baseMovementSpeed	Base movement speed
- int addMovementSpeed	Additional movement speed
- int evadeRate	Entity's evade rate
- int damageIncrease	Entity's damage increase
- int damageDecrease	Entity's damage decrease
- FaceDirection faceDirection	Entity's face direction
- Animation animation	Entity's animation
- HitBox hitBox	Entity's hitbox
- float attackRange	Entity's attack range
- String imgString	String for animation sprite image
- int standStillFrames	Number of stand still frames
- int walkFrames	Number of walk frames
- int normalAttackFrames	Number of attack frames
- int deadFrames	Number of dead frames
- ArrayList<Integer> frameMakeDamageNormalAttack	Arraylist that contain frames that can make damage in normal attack
- boolean canTakeDamage (= true)	Boolean for checking whether that can

	take damage
- boolean isDash	Boolean for checking whether that currently dashing
- float knockbackDistance	Entity's knockback distance
- ArrayList<Integer> dashFrame	Arraylist that contain frames that will dash
- int dashFrameCount	Number of dash frame count
- int animationWidth	Animation width
- int animationHeight	Animation height
- int hitBoxWidth	Hitbox width
- int hitBoxHeight	Hitbox height
- int hitBoxXOffset	Hitbox offset in axis X
- int hitBoxYOffset	Hitbox offset in axis Y
- ArrayList<SoundLoader> soundList	Arraylist that contain sound
- int damageReceived	Damage that attacker dealt

Methods

Name	Description
+ void setCommonStatus()	Set common status
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status
+ int getBaseMaxHp()	Getter for baseMaxHp
+ void setBaseMaxHp(int baseMaxHp)	Setter for baseMaxHp - Base max health can't below 0
+ int getAddMaxHp()	Getter for addMaxHp
+ void setAddMaxHp(int addMaxHp)	Sette for addMaxHp
+ int getCurrentHp()	Getter for currentHp
+ void setCurrentHp(int currentHp)	Setter for currentHp - Current health can't below 0 and can't above max health

+ int getBaseAtk()	Getter for baseAtk
+ void setBaseAtk(int baseAtk)	Setter for baseAtk - Base attack can't below 0
+ int getAddAtk()	Getter for addAtk
+ void setAddAtk(int addAtk)	Setter for addAtk
+ int getBaseDef()	Getter for baseDef
+ void setBaseDef(int baseDef)	Setter for baseDef - Base defense can't below 0
+ int getAddDef()	Getter for addDef
+ void setAddDef(int addDef)	Setter for addDef
+ float getBaseMovementSpeed()	Getter for baseMovementSpeed
+ void setBaseMovementSpeed(float baseMovementSpeed)	Setter for baseMovementSpeed - Base movement speed can't below 0
+ int getAddMovementSpeed	Getter for addMovementSpeed
+ void setAddMovementSpeed(int addMovementSpeed)	Setter for addMovementSpeed
+ int getEvadeRate()	Getter for evadeRate
+ void setEvadeRate(int evadeRate)	Setter for evade rate - Evade rate can't below 0
+ int getDamageIncrease()	Getter for damageIncrease
+ void setDamageIncrease(int damageIncrease)	Setter for damageIncrease
+ int getDamageDecrease()	Getter for damageDecrease
+ void setDamageDecrease()	Setter for damageDecrease
+ FaceDirection getFaceDirection()	Getter for faceDirection
+ void setFaceDirection(FaceDirection faceDirection)	Setter for faceDirection
+ float getAttackRange()	Getter for attackRange
+ void setAttackRange(float attackRange)	Setter for attackRange

+ Animation getAnimation()	Getter for animation
+ void setAnimation(Animation animation)	Setter for animation
+ HitBox getHitBox()	Getter for hitBox
+ void setHitBox(HitBox hitBox)	Setter for hitBox
+ String getImgString()	Getter for imgString
+ void setImgString()	Setter for imgString
+ int getStandStillFrames()	Getter for standStillFrames
+ void setStandStillFrames(int standStillFrames)	Setter for standStillFrames
+ int getWalkFrames()	Getter for walkFrames
+ void setWalkFrames(int walkFrames)	Setter for walkFrames
+ int getNormalAttackFrames()	Getter for normalAttackFrames
+ void setNormalAttackFrames(int normalAttackFrames)	Setter for normalAttackFrames
+ int getDeadFrames()	Getter for deadFrames
+ void setDeadFrames(int deadFrames)	Setter for deadFrames
+ ArrayList<Integer> getFrameMakeDamageNormalAttack()	Getter for frameMakeDamgeNormalAttack
+ void setFrameMakeDamageNormalAttack(ArrayList<Integer> frameMakeDamageNormalAttack)	Setter for frameMakeDamageNormalAttack
+ boolean canTakeDamage()	Getter for canTakeDamage
+ void setCanTakeDamage(boolean canTakeDamage)	Setter for canTakeDamage
+ float getKnockbackDistance()	Getter for knockbackDistance
+ void setKnockbackDistance(float knockbackDistance)	Setter for knockbackDistance

+ ArrayList<Integer> getDashFrame()	Getter for dashFrames
+ void setDashFrame(ArrayList<Integer> dashFrame)	Setter for dashFrames
+ int getDashFrameCount()	Getter for dashFrameCount
+ void setDashFrameCount(int dashFrameCount)	Setter for dashFrameCount
+ boolean isDash()	Getter for isDash
+ void setDash(boolean dash)	Setter for isDash
+ int getAnimationWidth()	Getter for animationWidth
+ void setAnimationWidth(int animationWidth)	Setter for animationWidth
+ int getAnimationHeight()	Getter for animationHeight
+ void setAnimationHeight(int animationHeight)	Setter for animationHeight
+ int getHitBoxWidth()	Getter for hitBoxWidth
+ void setHitBoxWidth(int hitBoxWidth)	Setter for hitBoxWidth
+ int getHitBoxHeight()	Getter for hitBoxHeight
+ void setHitBoxHeight(int hitBoxHeight)	Setter for hitBoxHeight
+ int getHitBoxWidth()	Getter for hitBoxWidth
+ void setHitBoxWidth(int hitBoxWidth)	Setter for hitBoxWidth
+ int getHitBoxHeight()	Getter for hitBoxHeight
+ void setHitBoxHeight(int hitBoxHeight)	Setter for hitBoxHeight
+ int getHitBoxXOffset()	Getter for hitBoxXOffset
+ void setHitBoxXOffset()	Setter for hitBoxXOffset
+ int getHitBoxYOffset()	Getter for hitBoxYOffset
+ void setHitBoxYOffset(int	Setter for hitBoxYOffset

hitBoxYOffset)	
+ ArrayList<SoundLoader> getSoundList()	Getter for soundList
+ void setSoundList(ArrayList<SoundLoader> soundList)	Setter for soundList
+ int getDamageReceived()	Getter for damageReceived
+ void setDamageReceived(int damageReceived)	Setter for damageReceived

5.3.4. Abstract Class BaseGiGee extends BaseMonster

- This class is used as a base class of normal monsters (called GiGee)

Constructors

Name	Description
+ BaseGiGee()	Constructor for BaseGiGee - super() - Set common status

Methods

Name	Description
+ void setCommonStatus()	Set common status - Set evade rate to LOW_EVADE_RATE - Set drop rate to GIGEE_DROP_RATE - Set coin drop to COIN_DROP_GIGEE - Set knockback chance to HIGH_KNOCKBACK_CHANCE - Set knockback distance to HIGH_KNOCKBACK_DISTANCE

5.3.5. Abstract Class BaseMonster extends BaseEntity

- This class is used as a base class of all monsters (normal monsters and boss)

Fields

Name	Description

- float normalAttackDelay	Time that can’t attack after normal attack
- float addAttackDelay	Additional attack delay
- float dropRate	Chance to drop health potion
- int coinDrop	Amount of coin drop
- float visionRange	Range of vision for the monster
- float walkTime	Time duration for monster to walk when idle
- float time	Time that use for check duration of walkTime and standTime
- float timeSinceLastAttack (= 10)	Time since last attack
- float standTime	Time duration for monster to stand still when idle
- float turningTime	Time duration for monster to turn around
- boolean isWalk	Boolean for checking whether that currently walking
- boolean canSee (= false)	Boolean for checking whether that monster can see player
- boolean canAttack (= false)	Boolean for checking whether that monster can attack player
- boolean isSpawn (= false)	Boolean for checking whether that monster is already spawn
- FaceDirection dashDirection (= null)	Direction of dashing
- boolean moreCoinDrop (= false)	Boolean for checking whether that monster drop more coins
- int knockbackChance	Knockback Chance
- boolean isSlowDownByThornPerk (= false)	Boolean for checking whether that monster is slow down by thorn perk
- ArrayList<String> CLOSE_RANGE_MONSTERS (= new ArrayList<>(Arrays.asList(“KleeG	Arraylist that contain all type of close range monster in the game

iGee", "PeasantGiGee")))	
- ArrayList<States> ATTACK_STATES = new ArrayList<>(Arrays.asList(MONSTER_NORMAL_ATTACK, BOSS_SKILL_ONE, BOSS_SKILL_TWO, BOSS_SKILL_THREE_STAND_STILL))	Arraylist that contain all states that doing attack action and can't move

Constructors

Name	Description
+ BaseMonster()	Constructor for BaseMonster - Set base status

Methods

Name	Description
+ void setBaseStatus()	Set base status for monster - Set additional max health to 0 - Set additional attack to 0 - Set additional defense to 0 - Set additional movement speed to 0 - Set damage increase to 0 - Set damage decrease to 0 - Set image string - Initialise new Arraylist for frameMakeDamgeNormalAttack, dashFrame - Set dash frame count to 0 - Load entities sound - Set face direction to randomStartDirection() - Set stand time to getNewRandomTime() - Set turning time to TURNING_DELAY - Set animation width to DEFAULT_ANIMATIO_WIDTH - Set animation height to DEFAULT_ANIMATION_HEIGHT - Set hitbox width to DEFAULT_HITBOX_WIDTH - Set hitbox height to

	DEFAULE_HITBOX_HEIGHT - Set hitbox offset in axis X to DEFAULT_HITBOX_X_OFFSET - Set hitbox offset in axis Y to DEFAULT_HITBOX_Y_OFFSET

+ void isAttacked(BaseEntity baseEntity, BaseScene scene, int extraDamagePercent)	Monster is attacked by player
	- Check if monster can evade this attack
	- If the monster can evade this attack, create text that says 'Miss'
	- If the monster can't evade this attack
	  - If player has Vampirism perk, activate vampirism perk
	  - Calculate damage
	  - If monster's current hp equal 0, monster dead

+ void hurt(BaseEntity attacker, BaseScene baseScene)	Monster gets damage - Play monster' hurt sound - Check if monster gets knockback
+ void dead(BaseEntity attacker, BaseScene baseScene)	Monster dead - Play monster's dead sound - Calculate coin drop from monster - If player has Treasure hunter perk tier 2.2, get more coin - Set player coin - Set state monster to MONSTER_DEAD - If monster's dead animation need to bounce off, set yVelocity to -18 * SCALE

+ void move(BaseScene scene)	Monster moves
	- If monster's state is not MONSTER_DEAD
	  - Check dash
	  - If monster's state is not in ATTACK_STATES
	  - If monster can see player, prepare to attack
	  - If monster is walking, check direction
	- If monster's state is

	MONSTER_DEAD and dead animation need to bounce off, call dash() - If monster is not jump, call checkFall() - If monster is jump, call jump()
- void checkFall(BaseScene scene, double hitBoxX, double hitBoxY, double width, double height)	Check if monster will fall - If 0.5 pixel below can move down, set yVelocity to 0 and set jump to true
- void jump(BaseScene scene, double hitBoxX, double hitBoxY, double animationX, double animationY, double width, double height)	Move monster in y axis - If yVelocity is less than or equal to 0 and the monster can move upwards, move the monster's position upwards - If yVelocity is more than 0 and the monster can't move downwards, find the floor and move the monster to stand on it and set jump to false
- void checkDirection(BaseScene scene, double hitBoxX, double hitBoxY, double animationX, double width, double height, float moveSpeed)	Call checkWalk() matching the face direction
- void checkWalk(FaceDirection faceDirection, BaseScene scene, double hitBoxX, double hitBoxY, double animationX, double width, double height, float moveSpeed)	Adjust direction and check if the monster can walk - If face direction is LEFT, multiply movespeed by -1 - If can walk, call walk() - If can’t walk, set time to getWalkTime() to make monster stop move
- boolean canWalk(FaceDirection faceDirection, double hitBoxX, double speed, double hitBoxY, double width, double height, int[][] lvlData)	Check if monster can walk - if it's possible to stand at the position after moving, return true; otherwise, return false
- void walk(double hitBoxX, double animationX, float speed, BaseScene scene)	Move monster in x axis - Move monster position - If the monster is BigBoy and there's fire in the stage, move the fire with the same speed as BigBoy

+ boolean canAttack(BaseEntity baseEntity)	Check if monster can attack player - Check distance in y axis. If more than 1 tile, return false; otherwise, return true - Check distance in x axis. If more than monster's attack range, return false; otherwise, return true
- boolean prepareToAttack(BaseScene scene, float moveSpeed)	Check if the monster needs to move to attack player - If the monster is boss and distance between boss and player more than 0.1 tiles, call checkMoveToPlayer() - If the monster can't attack right now and is not a boss, call checkMoveToPlayer() - If the monster can attack, change direction to face player
+ void attack(Player player, BaseScene, int extraDamagePercent)	Monster attacks - If can attack and player can take damage, attack player
- void checkDash(BaseScene scene)	Check if the monster dash - Call dash() matching the monster's type
- void dash(double distance, int times, BaseScene scene, boolean toPlayer, Player player)	Move monster - If dashFrameCount lower than time, increase dashFrameCount by 1 and move monster; otherwise, set dash to false, set dash frame count to 0 and set dash direction to null
+ boolean isPlayerInVisionRange(BaseScene scene)	Check if player in monster's vision range - If the monster is a Boss, return true; otherwise, check the distance based on the monster's vision range. If it's true, call checkAttack()
- boolean checkAttack(BaseScene scene)	Check if the monster can attack based on monster' type - If the monster is ThrowerGiGee, return canThrowerGiGeeAttack(); otherwise, return

	canOtherAttack()
- boolean canOtherAttack(FaceDirection direction, double getX, double getPlayerX, double Y, BaseScene scene)	Check if the monster can attack (not ThrowerGiGee) - Check if there is a tile between the player and the monster; if so, return false - If the monster is a close-range monster and there is an airTile below between the player and the monster, return false; otherwise, return true
- boolean canThrowerGiGeeAttack(BaseScene scene, double getX, double getY)	Check if the ThrowerGiGee can attack - Find a path for projectile attack to the player. If there is a path, return true; otherwise, return false
- void checkMoveToPlayer(Player player, float speed, BaseScene scene)	Call moveToPlayer() matching the face direction
- void moveToPlayer(FaceDirection toMoveDirection, double hitBoxX, double hitBoxY, double animationX, double width, double height, float speed, BaseScene scene)	Move monster to player - If the monster's face direction matches the direction to move, set state to MONSTER_WALK, and if canWalk(), call walk() - If the monster's face direction doesn't match the direction to move, and if the time is more than turningTime, make the monster turn around; otherwise, increase time by 1.0 / MOVE_FRAME
+ boolean isPlayerInAttackRange(Player player)	Check if player in monster’s attack range - If the monster is a Boss, return true; otherwise, check the distance based on the monster’s attack range
+ float getDeltaTilesX(Player player)	Check distance between monster and player in axis x
+ float getDeltaTilesY(Player player)	Check distance between monster and player in axis y
+ float getNormalAttackDelay()	Getter for normalAttackDelay

+ void setNormalAttackDelay(float baseAttackDelay)	Setter for normalAttackDelay - Normal attack delay can't below 0.5
+ float getAddAttackDelay()	Getter for addAttackDelay
+ void setAddAttackDelay(float addAttackDelay)	Setter for addAttackDelay
+ float getDropRate()	Getter for dropRate
+ void setDropRate(float dropRate)	Setter for dropRate
+ int getCoinDrop()	Getter for coinDrop
+ void setCoinDrop(int coinDrop)	Setter for coinDrop
+ float getVisionRange()	Getter for visionRange
+ void setVisionRange(float visionRange)	Setter for visionRange - Vision range can't below 1
+ float getWalkTime()	Getter for walkTime
+ void setWalkTime(float walkTime)	Setter for walkTime
+ float getNewRandomTime()	Return random time between MIN_WALKTIME and MAX_WALKTIME
+ float getTime()	Getter for time
+ void setTime(float time)	Setter for time
+ float getStandTime()	Getter for standTime
+ void setStandTime()	Setter for standTime
+ void setWalk()	Getter for isWalk
+ void setCanSee(boolean canSee)	Setter for canSee
+ void setCanAttack(boolean canAttack)	Setter for canAttack
+ boolean isPlayerAtRight(Player player)	Return true if the player is to the right of the monster; otherwise, return false
+ boolean isPlayerAtLeft(Player player)	Return true if the player is to the left of the monster; otherwise, return false
+ void setTurningTime(float turningTime)	Setter for turningTime

+ float getTimeSinceAttack()	Getter for timeSinceAttack
+ void setTimeSinceAttack(float timeSinceLastAttack)	Setter for timeSinceAttack
+ int getKnockbackChance()	Getter for knockbackChance
+ void setKnockbackChance(int knockbackChance)	Setter for knockbackChance
+ boolean isSpawn()	Getter for isSpawn
+ void setSpawn(boolean spawn)	Setter for isSpawn
+ FaceDirection getDashDirection()	Getter for dashDirection
+ void setDashDirection(FaceDirection dashDirection)	Setter for dashDirection
+ void setMoreCoinDrop(boolean moreCoinDrop)	Setter for moreCoinDrop
+ boolean isSlowdownByThornPerk()	Getter for isSlowDownByThornPerk
+ void setSlowdownByThornPerk(boolean slowdownByThornPerk)	Setter for isSlowDownByThornPerk

5.3.6. Abstract Class BasePerk implements Obtainable, Upgradable, Tradable, Perkable

- This class is used as a base class of all perks

Fields

Name	Description
- Tier tier (= PERK_TIER1)	Perk's tier
- Player player	Player
- int originalPrice	Base price
- int newPrice	Price after discount
- int upgradeCost	Price to use when upgrading perk
- Rarity rarity	Perk's rarity
- boolean hasActivated (= false)	Boolean for checking whether that perk has activated

- String UI_STRING (= “perks/” + this + “_UI.png”)	String for UI image
- String ICON_STRING (= “perks/” + this + “_Icon.gif”)	String for icon image

Constructors

Name	Description
+ BasePerk(Player player, Tier tier, Rarity rarity)	Constructor for BasePerk - Set player to player - Set tier to tier - Set start values to getTier() - Set rarity to rarity - Set original price to getRarity()

Methods

Name	Description
+ boolean canBuy(Player player)	Check if can buy perk - If player has a DiscountMasterPerk - Return true if the player hasn't already obtained this perk, has more coins than the new price, and the perk inventory isn't full - Else, return true if the player hasn't already obtained this perk, has more coins than the original price, and the perk inventory isn't full
+ void buy(Player player)	Buy perk - If player has a DiscountMasterPerk - Reduce player’s coin by newPrice - Else, reduce player’s coin by originalPrice - If player has a DiscountMasterPerk Tier 2.2 and buy fail - Play buying miss sound - Else, call collect() and play buying success sound
+ void collect(Player player)	Collect perk

	- Add this perk to player's obtain perks - Remove this perk from unobtained perks
+ boolean canUpgrade(Player player)	Check if can upgrade perk - If player has DiscountMasterPerk, set upgradePrice to PRICE_UPGRADE_PERCENT * newPrice otherwise set it to PRICE_UPGRADE_PERCENT * originalPrice - return true if player has more coin than upgradePrice and this perk has TIER1
+ void upgrade(Player player, Tier tier)	Upgrade perk - Deactivate this perk - Remove this perk from player's obtain perks - Set tier to tier - Decrease player's money by upgradePrice - Set start value to getTier() - Add this perk to player's obtain perks
+ *void setStartValues(Tier tier)*	Set start stat for some perks
+ String toString()	Return class name
+ Tier getTier()	Getter for tier
+ void setTier(Tier tier)	Setter for tier
+ int getOriginalPrice()	Getter for originalPrice
+ void setOriginalPrice(Rarity rarity)	Setter for originalPrice - Set original price based on rarity
+ int getNewPrice()	Getter for newPrice
+ void setNewPrice(int price)	Setter for newPrice - If player has DiscountMasterPerk, reduce price based on DiscountMasterPerk's discount percent
+ boolean hasActivated()	Getter for hasActivated

+ void setHasActivated(boolean activated)	Setter for hasActivated
+ String getIconString()	Getter for ICON_STRING
+ ImageView getImageAtRowColumn(Tier tier, int column)	Set viewport to specific perk's tier and column
+ Rarity getRarity()	Getter for rarity
+ void setRarity(Rarity)	Setter for rarity
+ Player getPlayer()	Getter for player
+ void setPlayer(Player player)	Setter for player
+ int getUpgradeCost()	Getter for upgradeCost
+ void setUpgradeCost()	Setter for upgradeCost

5.4. Package entities.monsters

5.4.1. Class ArcherGiGee extends BaseGiGee

- This class is used as a long-range GiGee that uses Missile

Constructors

Name	Description
+ ArcherGiGee(double spawnX, double spawnY)	Constructor for ArcherGiGee - Call super() - Set unique status with spawnX, spawnY

Methods

Name	Description
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status - Set base max health to ARCHER_HEALTH - Set current health equal to base max health - Set base attack to ARCHER_ATK - Set base defense to ARCHER_DEF - Set base movement speed to MEDIUM_MOVEMENT_SPEED

	- Set attack range to LONG_RANGE - Set vision range to MEDIUM_VISION - Set normal attack delay to ARCHER_ATK_DELAY - Set stand still frames to ARCHER_STAND_STILL_FRAMES - Set walk frames to ARCHER_WALK_FRAMES - Set normal attack frames to ARCHER_NORMAL_ATTACK_FRMAES - Set dead frames to ARCHER_DEAD_FRAMES - Add 19 to frameMakeDamageNormalAttack - Set animation to new animation - Set hitbox to new hitbox

5.4.2. Class BigBoy extends BaseBoss

- This class is used as a Boss that uses Fire, ShockWave and Stomping Attack

Fields

Name	Description
- double jumpDistance	Jump distance for skill 2

Constructors

Name	Description
+ BigBoy(double spawnX, double spawnY)	Constructor for BigBoy - Call super() - Set unique status with spawnX, spawnY

Methods

Name	Description
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status - Set base max health to BIGBOY_HP - Set current health equal to base

	max health - Set base attack to BIGBOY_ATK - Set base defense to BIGBOY_DEF - Set base movement speed to MEDIUM_MOVEMENT_SPEED - Set cooldown skill 1 to BIGBOY_SKILL_ONE_COOLDOWN - Set cooldown skill 2 to BIGBOY_SKILL_TWO_COOLDOWN - Set cooldown skill 3 to BIGBOY_SKILL_THREE_COOLDOWN - Set skill 1 attack range to BIGBOY_SKILL_ONE_ATTACK_RANGE - Set skill 2 attack range to BIGBOY_SKILL_TWO_ATTACK_RANGE - Set skill 3 attack range to BIGBOY_SKILL_THREE_ATTACK_RANGE - Set stand still frames to BIGBOY_STAND_STILL_FRAMES - Set walk frames to BIGBOY_WALK_FRAMES - Set dead frames to BIGBOY_DEAD_FRAMES - Set face direction to LEFT - Add 5 to frameMakeDamageSkillOne - Add 3 to dashFrame - Add 3 to frameMakeDamageSkillTwo - Add 4 to frameMakeDamageSkillThree - Set animation width to BIGBOY_ANIMATION_WIDTH - Set animation height to BIGBOY_ANIMATION_HEIGHT - Set hitbox width to BIGBOY_HITBOX_WIDTH - Set hitbox height to BIGBOY_HITBOX_HEIGHT - Set hitbox offset in axis X to BIGBOY_HITBOX_X_OFFSET

	- Set hitbox offset in axis Y to BIGBOY_HITBOX_Y_OFFSET - Set animation to new animation - Set hitbox to new hitbox
+ double getJumpDistance()	Getter for jumpDistance
+ void setJumpDistance(double jumpDistance)	Setter for jumpDistance

5.4.3. Class IceThrowerGiGee extends ThrowerGiGee

- This class is used as an another type of ThrowerGiGee that throws IceThrowerBomb

Constructors

Name	Description
+ IceThrowerGiGee(double spawnX, double spawnY)	Constructor for IceThrowerGiGee - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status - Set base max health to THROWER_HEALTH - Set current health equal to base max health - Set base attack to ICE_THROWER_ATK - Set base defense to THROWER_DEF - Set base movement speed to LOW_MOVEMENT_SPEED - Set attack range to LONG_RANGE - Set vision range to MEDIUM_VISION - Set normal attack delay to THROWER_ATK_DELAY - Set stand still frames to ICETHROWERGIGEE_STAND_STILL_FRAMES - Set walk frames to ICETHROWERGIGEE_WALK_FRAMES

	- Set normal attack frames to ICETHROWERGIGEE_NORMAL_ATTACK_FRMAES - Set dead frames to ICETHROWERGIGEE_DEAD_FRAMES - Add 13 to frameMakeDamageNormalAttack - Set animation to new animation - Set hitbox to new hitbox

5.4.4. Class KleeGiGee extends BaseGiGee

- This class is used as a short-range GiGee that sacrifices himself to deal damage, creating KleeBomb.

Constructors

Name	Description
+ KleeGiGee(double spawnX, double spawnY)	Constructor for KleeGiGee - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status - Set base max health to KLEE_HEALTH - Set current health equal to base max health - Set base attack to KLEE_ATK - Set base defense to KLEE_DEF - Set base movement speed to HIGH_MOVEMENT_SPEED - Set attack range to CLOSED_RANGE - Set vision range to MEDIUM_VISION - Set normal attack delay to KLEE_ATK_DELAY - Set turning time to KLEE_TURNING_DELAY - Set stand still frames to KLEEGIGEE_STAND_STILL_FRAMES - Set walk frames to

	KLEEGIGEE_WALK_FRAMES - Set normal attack frames to KLEEGIGEE_NORMAL_ATTACK_FRMAES - Set dead frames to KLEEGIGEE_DEAD_FRAMES - Set animation to new animation - Set hitbox to new hitbox
+ void activateBomb(BaseScene scene)	Klee activate bomb - Use AnimationTimer to timer a bomb that will activate in KLEEGIGEE_BOMB_TIME - During the AnimationTimer run, blink every 0.3 seconds

5.4.5. Class PeasantGiGee extends BaseGiGee

- This class is used as a short-range GiGee that deals damage by his sword.

Constructors

Name	Description
+ PeasantGiGee(double spawnX, double spawnY)	Constructor for PeasantGiGee - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status - Set base max health to PEASANT_HEALTH - Set current health equal to base max health - Set base attack to PEASANT_ATK - Set base defense to PEASANT_DEF - Set base movement speed to MEDIUM_MOVEMENT_SPEED - Set attack range to CLOSED_RANGE - Set vision range to SHORT_VISION - Set normal attack delay to PEASANT_ATK_DELAY

	- Set stand still frames to PEASANTGIGEE_STAND_STILL_FRAMES - Set walk frames to PEASANTGIGEE_WALK_FRAMES - Set normal attack frames to PEASANTGIGEE_NORMAL_ATTACK_FRMAES - Set dead frames to PEASANTGIGEE_DEAD_FRAMES - Add 8 to frameMakeDamageNormalAttack - Add 7 to dashFrame - Set animation to new animation - Set hitbox to new hitbox

5.4.6. Class ThrowerGiGee extends GiGee

- This class is used as a long-range GiGee that uses ThrowerBomb (can attack target that is on the different floor)

Constructors

Name	Description
+ ThrowerGiGee(double spawnX, double spawnY)	Constructor for ThrowerGiGee - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus(double spawnX, double spawnY)	Set unique status - Set base max health to THROWER_HEALTH - Set current health equal to base max health - Set base attack to THROWER_ATK - Set base defense to THROWER_DEF - Set base movement speed to LOW_MOVEMENT_SPEED - Set attack range to LONG_RANGE - Set vision range to

	MEDIUM_VISION - Set normal attack delay to THROWER_ATK_DELAY - Set stand still frames to THROWERGIGEE_STAND_STILL_FRAMES - Set walk frames to THROWERGIGEE_WALK_FRAMES - Set normal attack frames to THROWERGIGEE_NORMAL_ATTACK_FRMAES - Set dead frames to THROWERGIGEE_DEAD_FRAMES - Add 13 to frameMakeDamageNormalAttack - Set animation to new animation - Set hitbox to new hitbox

5.5. Package entities.player

5.5.1. Class Player extends BaseEntity

- This class is used as a player

Fields

Name	Description
- Player instances	Player himself
- BaseClass playerClass	Player's class
- float baseAttackSpeed	Player's Attack Speed at the beginning
- int addAttackSpeed	Player's additional Attack Speed
- int critRate	Player's Critical Rate
- int critDamage	Player's Critical Damage
- int coins	Player's coin
- String uiString	String for UI Image
- String inventoryUIString	String for Inventory UI Image
- int specialAttackFrames	Amount of frames that player's special attack uses
- int dashFrames;	Amount of frames that the dash uses

- ArrayList<BasePerk> obtainPerks;	ArrayList of Perks obtained by Player
- ArrayList<BasePerk> notObtainPerks;	ArrayList of Perks that are not obtained
- ArrayList<BaseMonster> monstersInStage	ArrayList of Monsters that are in the stage
- ArrayList<BaseObject> objectsInStage	ArrayList of Objects that are in the stage
- ArrayList<Integer> frameMakeDamageSpecialAttack	Arraylist for contain frame that can make damage in Special Attack
- int attackCombo	Number of attack combo
- float timeSinceLastAttack	Time duration starting when the attack is launched
- float timeCanSecondAttack	Time duration that still can second attack
- boolean isSecondAttack	Boolean to check whether player use the second attack or not
- boolean canInteract (= false)	Boolean to check whether player can interact or not
- BaseObject objectThatInteractWith (= null)	Object that player interacts with
- int remainingDash	Number of dash remaining
- float dashDistance	Distance the dash can go
- int addDashDistance	Extra distance the dash can go
- int dashTime	Time that dash uses
- boolean canDash (= true)	Boolean to check whether player can dash or not
- float specialAttackCooldown	Cooldown for Special Attack
- boolean isSpecialAttackInCooldown (= false)	Boolean to check whether the Special Attack is in cooldown or not
- boolean isTreasureHunterActivate (= false)	Boolean to check whether TreasureHunter Perk is activated or not

Constructors

Name	Description
+ Player()	Constructor for Player: - playerClass = new Anonymous() - spawnX = SPAWN_X - spawnY = SPAWN_Y
+ Player(BaseClass baseClass)	Constructor for Player: - spawnX = SPAWN_X - spawnY = SPAWN_Y
+ Player(BaseClass baseClass, double spawnX, double spawnY)	Constructor for Player - setStatsFromClass(baseClass) - setFramesFromClass(baseClass) - Set coin to START_COINS - Set new Animation - Set new HitBox - Set face direction to RIGHT - Initialize monstersInStage and objectsInStage - setSoundList(loadEntititesSound(this)) - Initialize obtainPerks - Set notObtainPerks to generateAllPerks(this)

Methods

Name	Description
+ void setFramesFromClass(BaseClass baseClass)	- Set imgString, uiString, inventoryUIString, standStillFrames, walkFrames, normalAttackFrames, specialAttackFrames, dashFrames, deadFrames, attackCombo, timeCanSecondAttack, frameMakeDamageNormalAttack, frameMakeDamageSpecialAttack, knockbackDistance, animationWidth, animationHeight, hitBoxWidth, hitBoxHeight, hitBoxXOffset and hitBoxYOffset to baseClass's
+ void setStatsFromClass(BaseClass baseClass)	- Set player class to baseClass - Set baseMaxHp to baseClass's hp - Set currentHp to baseMaxHp

	- Set addMaxHp to 0 - Set baseAtk to baseClass's atk - Set addAtk to 0 - Set baseDef to baseClass's def - Set addDef to 0 - Set baseMovementSpeed to baseClass's baseMovementSpeed - Set addMovementSpeed to 0 - Set addDashDistance to 0 - Set damageIncrease, damageDecrease, attackRange and baseAttackSpeed to baseClass's - Set addAttackSpeed to 0 - Set critRate, critDamage, maxDash to baseClass's - Set remainingDash to getMaxDash() - Set dashDistance to DEFAULT_DASH_DISTANCE - Set dashTime to DEFAULT_DASH_TIME - Set specialAttackCooldown to baseClass's specialAttackCooldown
+ void setCommonStatus()	Set common status based on class
+ void setUniqueStatus(double spawnX, spawnY)	Set unique status based on class
+ void move(BaseScene scene)	Player moves - Set border left and right - Check dash - If isMoveLeft and not isMoveRight, call moveLeft(); otherwise, if isMoveRight and not isMoveLeft, call moveRight() - If player is not jump, call checkFall() - If player is jump, call moveBorderYAxis(), checkStateForJump() and jump() - If scene now is ClassSelectionScene or MarketScene, call actionForRestScene()
- void moveBorderYAxis(BaseScene scene, double animationY)	Move border y axis - If the player moves near the border, move the scene in the opposite direction to the player
- void checkStateForJumping()	Switch state to match jumping - Increase yVelocity by GRAVITY - If the state is

	PLAYER_STAND_STILL, PLAYER_WALK, PLAYER_JUMP_UP, or PLAYER_JUMP_DOWN, and yVelocity is above 0, set state to PLAYER_JUMP_DOWN; otherwise, set state to PLAYER_JUMP_UP
- void jump(BaseScene scene, double hitBoxX, double hitBoxY, double animationY, double width, double height)	Move player in y axis - If yVelocity is less than or equal to 0 and the player can move upwards, move the player's position upwards - If yVelocity is more than 0 and the player can't move downwards, find the floor and move the player to stand on it and set jump to false
- void checkFall(BaseScene scene, double hitBoxX, double hitBoxY, double width, double height)	Check if player will fall - If isMoveLeft or isMoveRight, play walk sound - If 0.5 pixel below can move down, set yVelocity to 0 and set jump to true
- void moveRight(BaseScene scene, double animationX, double hitBoxX, double hitBoxY, double width, double height, double moveSpeed)	Move player to right - If player can move right, move player to right - If near border right and border right is not more than the background width, move the border to the left
- void moveLeft(BaseScene scene, double animationX, double hitBoxX, double hitBoxY, double width, double height, double moveSpeed)	Move player to left - If player can move left, move player to left - If near border left and border left is not less than 0, move the border to the right
- void actionForRestScene(BaseScene scene)	Check player near right side of the scene and change scene - If player move near right side of the scene and player's class is not Anonymous, check these things otherwise do nothing - If scene is a ClassSelectionScene, do actionForClassSelectionScene(scene) - If scene is a MarketScene, do actionForMarketScene(scene) - If scene is a BossScene and there are no monsters, do actionForEmptyBossScene(scene)

- void actionForEmptyBossScene(BaseScene scene)	Use runLater and inside of it, calling stopAllThreads() then set root for stage to new FightScene() and resetBossScene
- void actionForMarketScene(BaseScene scene)	stopAllThreads() then use runLater and inside of it. If there is an object that is a BlackSmith, call setStop()'s Blacksmith. After the for loop, set root for stage to new BossScene and resetMarketScene
- void actionForClassSelectionScene(BaseScene scene)	stopAllThreads() then use runLater and inside of it. If there is an object that is VendingMachine call setStop()'s VendingMachine. After the for loop, set root for stage to new FightScene
+ boolean canWarpToNearestMonster()	Check if player can warp to monster - If has monster that distance below DAGGER_SPECIAL_ATTACK_WARP_DISTANCE, return true
+ void warpToNearestMonster(BaseScene scene)	Warp to nearest monster - Check distance between player and monster for every monsters - Move player to nearest monster and call setScenePosition() and move playerUI to match with screen
+ boolean canAttack(BaseEntity baseEntity)	Check if player can attack monster - If monster's state is MONSTER_DEAD or MONSTER_SPAWN, return false - Check distance in y axis. If more than 1 tile, return false; otherwise - Check distance in x axis. If lower than monster's attack range, return true; otherwise, return false
+ void attack(Player player, BaseScene scene, int extraDamagePercent)	Player attacks - For every monster, check if the player can attack that monster, and if so, the monster is attacked
+ void isAttacked(BaseEntity baseEntity, BaseScene scene, int extraDamagePercent)	Player is attacked by monster - Check if monster can evade this attack - If the monster can evade this attack, create text that says 'Miss' - If the monster can't evade this attack - Calculate damage - If the damage is more than the

	current health, check if the player has the UndeadPerk and it's not in cooldown, then activate UndeadPerk; otherwise, reduce current health by damage - If player has ThornPerk and randomChace() < reflect chance, call reflect() - If current health is 0, if not has UndeadPerk, call dead(); otherwise, if UndeadPerk in cooldown, call dead(); otherwise, call hurt()
+ void hurt(BaseEntity attacker, BaseScene scene)	Player gets damage - Play player' hurt sound - Check if monster gets knockback
+ void dead(BaseEntity attacker, BaseScene scene)	Player dead - Play player' dead sound - Set state to PLAYER_DEAD - Stop all thread - Stop all background songs - Change scene to DeathScene
- void reflect(BaseEntity attacker, BaseScene scene)	Reflect damage back to monster - Calculate reflection damage and reduce monster health by this damage - If player has ThornPerk Tier 2.2, slow monster - If monster health is 0, call monster.dead(); otherwise, call monster.hurt()
- boolean isNearBorderLeft()	Check if player is in left side of the screen
- boolean isNearBorderRight()	Check if player is in right side of the screen
- void checkDash(BaseScene scene)	Check if player dash - Call dash() if isDash
- void dash(double distance, int times, FaceDirection faceDirection, BaseScene scene)	Move player - If dashFrameCount lower than time, increase dashFrameCount by 1 and move player and if near border move border; otherwise, set dash frame to 0 and make AnimationTimer to setCanDash() back to true in 0.1 seconds

+ void resetPlayer()	Reset player Set instance to null
+ Player getPlayer()	Getter for player - If instance is null, initialise new player; otherwise, return instance
+ ArrayList<BasePerk> getObtainPerks()	Getter for obtainPerks
+ ArrayList<BaseMonster> getMonstersInStage()	Getter for monsterInStage
+ ArrayList<BaseObject> getObjectsInStage()	Getter for objectsInStage
+ float getBaseAttackSpeed()	Getter for baseAttackSpeed
+ void setBaseAttackSpeed(float baseAttackSpeed)	Setter for baseAttackSpeed - Base attack speed can't below 1
+ int getCritRate()	Getter for critRate
+ void setCritRate(int critRate)	Setter for critRate - Critical rate can't below 0
+ int getCritDamage()	Getter for critDamage
+ void setCritDamage(int critDamage)	Setter for critDamage - Critical damage can't below 0
+ BaseClass getPlayerClass()	Getter for playerClass
+ void setPlayerClass(BaseClass playerClass)	Setter for playerClass
+ int getCoins()	Getter for coins
+ void setCoins(int coins)	Setter for coins - Coins can't below 0
+ int getAddAttackSpeed()	Getter for addAttackSpeed
+ void setAttackSpeed(int addAttackSpeed)	Setter for addAttackSpeed
+ ArrayList<BasePerk> getNotObtainPerks()	Getter for notObtainPerks
+ int getSpecialAttackFrames()	Getter for specialAttackFrames
+ void setSpecialAttackFrames(int specialAttackFrames)	Setter for specialAttackFrames

+ int getAttackCombo()	Getter for attackCombo
+ void setAttackCombo(int attackCombo)	Setter for attackCombo
+ float getTimeSinceLastAttack()	Getter for timeSinceLastAttack
+ void setTimeSinceLastAttack(float timeSinceLastAttack)	Setter for timeSinceLastAttack
+ float getTimeCanSecondAttack()	Getter for timeCanSecondAttack
+ void setTimeCanSecondAttack(float timeCanSecondAttack)	Setter for timeCanSecondAttack
+ boolean isSecondAttack()	Getter for isSecondAttack
+ void setIsSecondAttack(boolean isSecondAttack)	Setter for isSecondAttack
+ String getUiString()	Getter for uiString
+ void setUiString(String uiString)	Setter for uiString
+ String getInventoryUIString()	Getter for inventoryUI
+ void setUIInventoryString(String inventoryUIString)	Setter for inventoryUI
+ boolean canInteract()	Getter for canInteract
+ void setCanInteract(boolean canInteract)	Setter for canInteract
+ BaseObject getObjectThatInteractWith()	Getter for objectThatInteractWith
+ void setObjectThatInteractWith(BaseObject objectThatInteractWith)	Setter for objectThatInteractWith
+ int getRemainingDash()	Getter for remainingDash
+ void setRemainingDash(int remainingDash)	Setter for remainingDash
+ int getMaxDash()	Getter for maxDash
+ void setMaxDash(int maxDash)	Setter for maxDash
+ float getDashDistance()	Getter for dashDistance

+ void setDashDistance(float dashDistance)	Setter for dashDistance
+ boolean canDash()	Getter for canDash
+ void setCanDash(boolean canDash)	Setter for canDash
+ int getDashFrames()	Getter for dashFrames
+ void setDashFrames(int dashFrames)	Setter for dashFrames
+ ArrayList<Integer> getFrameMakeDamageSpecialAttack()	Getter for frameMakeDamageSpecialAttack
+ void setFrameMakeDamageSpecialAttack(ArrayList<Integer> frameMakeDamageSpecialAttack)	Setter for frameMakeDamageSpecialAttack
+ int getDashTime()	Getter for dashTime
+ void setDashTime(int dashTime)	Setter for dashTime
+ float getSpecialAttackCooldown()	Getter for specialAttackCooldown
+ void setSpecialAttackCooldown(float specialAttackCoolDown)	Setter for specialAttackCooldown
+ boolean isSpecialAttackInCooldown()	Getter for isSpecialAttackInCooldown
+ void setSpecialAttackInCooldown(boolean specialAttackInCooldown)	Setter for isSpecialAttackInCooldown
+ boolean canGetDoubleReward()	Getter for isTreasureHunterActivate
+ void setTreasureHunterActivate(boolean treasureHunterActivate)	Setter for isTreasureHunterActivate
+ int getAddDashDistance()	Getter for addDashDistance
+ void setAddDashDistance(int addDashDistance)	Setter for addDashDistance

5.6. Package entities.player.classes

5.6.1. Class Anonymous extends BaseClass

- This class is used as a sample class, appearing only in ClassSelectionScene

Constructors

Name	Description
+ Anonymous()	Constructor for Anonymous - Set unique status

Methods

Name	Description
+ void setUniqueStatus()	Set unique status - Set stand still frames to ANONYMOUS_STAND_STILL_FRAMES - Set walk frames to ANONYMOUS_WAK_FRAMES

5.6.2. Class Bow extends BaseClass

- This class is used as one of the classes that the player can select. It's a long-range class using a bow as a weapon.

Constructors

Name	Description
+ Bow()	Constructor for Bow - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus()	Set unique status - Set attack to LOW_ATK - Set movement speed to MEDIUM_MOVEMENT_SPEED - Set evade rate to HIGH_EVADE_RATE - Set attack range to LONG_RANGE - Set attack speed to BOW_NORMAL_ATTACK_SPEE

	D_MULTIPLIER - Set stand still frames to BOW_STAND_STILL_FRAMES - Set walk frames to BOW_WALK_FRAMES - Set normal attack frames to BOW_NORMAL_ATTACK_FRAMES - Set special attack frames to BOW_SPECIAL_ATTACK_FRAMES - Set special attack cooldown to BOW_SPECIAL_ATTACK_COOLDOWN - Set dash frames to BOW_DASH_FRAMES - Set dead frames to BOW_DEAD_FRAMES - Set attack combo to 1 - Set time can second attack to 0 - Set knockback distance to MEDIUM_KNOCKBACK_DISTANCE

5.6.3. Class BroadSword extends BaseClass

- This class is used as one of the classes that the player can select. It's a short-range class using a big sword as a weapon.

Constructors

Name	Description
+ BroadSword()	Constructor for BroadSword - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus()	Set unique status - Set health to HIGH_HP - Set attack to HIGH_ATK - Set defense to HIGH_DEF - Set movement speed to MEDIUM_MOVEMENT_SPEED - Set attack range to MID_RANGE - Set attack speed to

	BROADSWORD_NORMAL_ATTACK_SPEED_MULTIPLIER - Set stand still frames to BROADSWORD_STAND_STILL_FRAMES - Set walk frames to BROADSWORD_WALK_FRAMES - Set normal attack frames to BROADSWORD_NORMAL_ATTACK_FRAMES - Set special attack frames to BROADSWORD_SPECIAL_ATTACK_FRAMES - Set special attack cooldown to BROADSWORD_SPECIAL_ATTACK_COOLDOWN - Set dash frames to BROADSWORD_DASH_FRAMES - Set dead frames to BROADSWORD_DEAD_FRAMES - Set attack combo to 2 - Set time can second attack to TIME_CAN_SECOND_ATTACK - Add 4 and 12 to frameMakeDamageNormalAttack - Add 2 to frameMakeDamageSpecialAttack - Set knockback distance to LOW_KNOCKBACK_DISTANCE - Set max dash to LOW_MAX_DASH - Set animation width to BROADSWORD_ANIMATION_WIDTH - Set animation height to BROADSWORDBROADSWORD_ANIMATION_HEIGHT - Set hitbox width to BROADSWORD_HITBOX_WIDTH - Set hitbox height to BROADSWORD_HITBOX_HEIGHT - Set hit box offset in axis X to BROADSWORD_X_OFFSET - Set hit box offset in axis Y to

	BROADSWORD_Y_OFFSET

5.6.4. Class Dagger extends BaseClass

- This class is used as one of the classes that the player can select. It's a short-range class using two-handed daggers as a weapon.

Constructors

Name	Description
+ Dagger()	Constructor for Dagger - Call super() - Set unique status

Methods

Name	Description
+ void setUniqueStatus()	Set unique status - Set attack to LOW_ATK - Set movement speed to HIGH_MOVEMENT_SPEED - Set attack speed to DAGGER_NORMAL_ATTACK_SPEED_MULTIPLIER - Set critical rate to HIGH_CRIT_RATE - Set critical damage to HIGH_CRIT_DAMAGE - Set stand still frames to DAGGER_STAND_STILL_FRAMES - Set walk frames to DAGGER_WALK_FRAMES - Set normal attack frames to DAGGER_NORMAL_ATTACK_FRAMES - Set special attack frames to DAGGER_SPECIAL_ATTACK_FRAMES - Set special attack cooldown to DAGGER_SPECIAL_ATTACK_COOLDOWN - Set dash frames to DAGGER_DASH_FRAMES - Set dead frames to DAGGER_DEAD_FRAMES - Set attack combo to 2

	- Set time can second attack to TIME_CAN_SECOND_ATTACK - Add 1 and 3 to frameMakeDamageNormalAttack - Add 2 to frameMakeDamageSpecialAttack - Set knockback distance to MEDIUM_KNOCKBACK_DISTANCE

5.7. Package entities.player.perks

5.7.1. Class BerserkPerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase atk when player has low hp

Constructors

Name	Description
+ BerserkPerk(Player player, Tier tier)	Constructor for BerserkPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true in 3 cases 1. This perk has TIER1 and player has hp not more than BERSERK_1_CRITERIA 2. This perk has TIER2_1 and player has hp not more than BERSERK_2_1_CRITERIA 3. This perk has TIER2_2 and player has hp not more than BERSERK_2_2_CRITERIA
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get addAtk += BERSERK_1_ADDATK if this perk has TIER1 2. Player will get addAtk += BERSERK_2_1_ADDATK if this perk has TIER2_1 3. Player will get addAtk +=

	BERSERK_2_2_ADDATK if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk has been activated, deactivate this perk and 1. Player will get addAtk -= BERSERK_1_ADDATK if this perk has TIER1 2. Player will get addAtk -= BERSERK_2_1_ADDATK if this perk has TIER2_1 3. Player will get addAtk -= BERSERK_2_2_ADDATK if this perk has TIER2_2
+ void setStartValues(Tier tier)	Do nothing

5.7.2. Class DiscountPerk extends BasePerk

- This class is used as a RARE perk for players. It can decrease the perk's price in the market

Fields

Name	Description
- float chance	The chance of being failed when buying a perk

Constructors

Name	Description
+ DiscountMasterPerk(Player player, Tier tier)	Constructor for DiscountMasterPerk - Call super(player, tier, RARE)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Do nothing
+ void activatePerk(Player player)	Do nothing
+ void deactivatePerk(Player player)	Do nothing
+ void setStartValues(Tier tier)	If tier is TIER2_2, set chance to DISCOUNT_2_2_FAILCHANCE otherwise set chance to DISCOUNT_NORMAL_FAILCHANCE

+ float getChance()	Return chance

5.7.3. Class ExtrovertPerk extends BasePerk

- This class is used as a UNCOMMON perk for players. It can increase atk if there are many monsters around

Fields

Name	Description
- float IN_RANGE_TILES_X (= EXTROVERT_TILES_X)	The amount of tiles in X-Axis
- float IN_RANGE_TILES_Y (= EXTROVERT_TILES_Y)	The amount of tiles in Y-Axis
- int monsterCountBefore	Amount of monsters at the last time
- int monsterCountAfter	Amount of monsters at the present time

Constructors

Name	Description
+ ExtrovertPerk(Player player, Tier tier)	Constructor for ExtrovertPerk - Call super(player, tier, UNCOMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	- Set monsterCountBefore to monsterCountAfter - Set monsterCountAfter to the amount of monsters around - Return true when monsterCountAfter is not less than EXTROVERT_CRITERIA otherwise return false
+ void activatePerk(Player player)	- Giving delta = monsterCountAfter - monsterCountBefore - If this perk has TIER1, increase addAtk by EXTROVERT_1_ADDATKPERMONSTER * delta - If this perk has TIER2_1, increase addAtk by EXTROVERT_2_1_ADDATKPE

	RMONSTER * delta - If this perk has TIER2_2, increase addAtk by EXTROVERT_2_2_ADDATKPERMONSTER * delta
+ void deactivatePerk(Player player)	- Giving delta = monsterCountBefore - monsterCountAfter - If this perk has TIER1, decrease addAtk by EXTROVERT_1_ADDATKPERMONSTER * delta - If this perk has TIER2_1, decrease addAtk by EXTROVERT_2_1_ADDATKPERMONSTER * delta - If this perk has TIER2_2, decrease addAtk by EXTROVERT_2_2_ADDATKPERMONSTER * delta
+ void setStartValues(Tier tier)	Set monsterCountBefore and monsterCountAfter to 0 and if this perk has TIER2_2, decrease addAtk by EXTROVERT_2_2_MINUSATKEXTRA

5.7.4. Class FatalAttackPerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase critical damage

Constructors

Name	Description
+ FatalAttackPerk(Player player, Tier tier)	Constructor for FatalAttackPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get critDmg +=

	FATAL_1_CRITDAMAGE if this perk has TIER1 2. Player will get critDmg += FATAL_2_1_CRITDAMAGE if this perk has TIER2_1 3. Player will get critDmg += FATAL_2_2_CRITDAMAGE but get critRate -= FATAL2_2_MINUSCRITRATE if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk has been activated, deactivate this perk and + Player will get critDmg -= FATAL_1_CRITDAMAGE if this perk has TIER1 + Player will get critDmg -= FATAL_2_1_CRITDAMAGE if this perk has TIER2_1 + Player will get critDmg -= FATAL_2_2_CRITDAMAGE but get critRate += FATAL2_2_MINUSCRITRATE if this perk has TIER2_2
+ void setStartValues(Tier tier)	Do nothing

5.7.5. Class FortifyPerk extends BasePerk

- This class is used as a COMMON perk for players. It can decrease damage received from all monsters

Constructors

Name	Description
+ FortifyPerk(Player player, Tier tier)	Constructor for FortifyPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get damageDecrease

	+= FORTIFY_1_ADDDAMAGEDECREASE if this perk has TIER1 2. Player will get damageDecrease += FORTIFY_2_1_ADDDAMAGEDECREASE if this perk has TIER2_1 3. Player will get damageDecrease += FORTIFY_2_2_ADDDAMAGEDECREASE and all monsters in the stage will get attackDelay += FORTIFY_2_2_ADDATKDELAY if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get damageDecrease -= FORTIFY_1_ADDDAMAGEDECREASE if this perk has TIER1 2. Player will get damageDecrease -= FORTIFY_2_1_ADDDAMAGEDECREASE if this perk has TIER2_1 3. Player will get damageDecrease -= FORTIFY_2_2_ADDDAMAGEDECREASE and all monsters in the stage will get attackDelay -= FORTIFY_2_2_ADDATKDELAY if this perk has TIER2_2
+ void setStartValues(Tier tier)	Do nothing

5.7.6. Class IntrovertPerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase atk when there are low monsters nearby

Fields

Name	Description
- float IN_RANGE_TILES_X (= INTROVERT_TILES_X)	The amount of tiles in X-Axis
- float IN_RANGE_TILES_Y (= INTROVERT_TILES_Y)	The amount of tiles in Y-Axis

- int monsterCount	The amount of monsters nearby
- int latestValue	The last value that is added

Constructors

Name	Description
+ IntrovertPerk(Player player, Tier tier)	Constructor for IntrovertPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Count monsters around players and set it to monsterCount then return true
+ void activatePerk(Player player)	- Assign toAddValue - If this perk has TIER_1, toAddValue = INTROVERT_1_BASEADDATK - INTROVERT_1_MINUSADDATKPERMONSTER * monsterCount (toAddValue must not be lower than 0) - If this perk has TIER2_1, toAddValue = INTROVERT_2_1_BASEADDATK - INTROVERT_2_1_MINUSADDATKPERMONSTER * monsterCount (toAddValue must not be lower than 0) - If this perk has TIER_2_2, toAddValue = INTROVERT_2_2_BASEADDATK - INTROVERT_2_2_MINUSADDATKPERMONSTER * monsterCount - If latestValue is not the same as toAddValue, addAtk += toAddValue - latestValue and set latestValue to toAddValue
+ void deactivatePerk(Player player)	addAtk -= latestValue

+ void setStartValues(Tier tier)	Set monsterCount and latestValue to 0

5.7.7. Class JukeMasterPerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase evade rate

Constructors

Name	Description
+ JukeMasterPerk(Player player, Tier tier)	Constructor for JukeMasterPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get evadeRate += JUKE_1_ADDEVADERATE if this perk has TIER1 2. Player will get damageDecrease += JUKE_2_1_ADDEVADERATE if this perk has TIER2_1 3. Player will get damageDecrease += JUKE_2_2_ADDEVADERATE and all monsters in the stage will get damageDecrease -= JUKE_2_2_MINUSDAMAGEDECREASE if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get evadeRate -= JUKE_1_ADDEVADERATE if this perk has TIER1 2. Player will get damageDecrease -= JUKE_2_1_ADDEVADERATE if this perk has TIER2_1 3. Player will get damageDecrease -= JUKE_2_2_ADDEVADERATE and all monsters in the stage will get damageDecrease +=

	JUKE_2_2_MINUSDAMAGEDECREASE if this perk has TIER2_2
+ void setStartValues(Tier tier)	Do nothing

5.7.8. Class LuckyManPerk extends BasePerk

- This class is used as a RARE perk for players. It can give a chance of getting a TIER 2 Perk in a chest or in MarketScene

Constructors

Name	Description
+ LuckyManPerk(Player player, Tier tier)	Constructor for LuckyManPerk - Call super(player, tier, RARE)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	Do nothing
+ void deactivatePerk(Player player)	Do nothing
+ void setStartValues(Tier tier)	Do nothing

5.7.9. Class PerfectionistPerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase atk when player has much hp

Constructors

Name	Description
+ PerfectionistPerk(Player player, Tier tier)	Constructor for PerfectionistPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true in 3 cases 1. This perk has TIER1 and player

	has hp not less than PERFECTION_1_CRITERIA 2. This perk has TIER2_1 and player has hp not less than PERFECTION_2_1_CRITERIA 3. This perk has TIER2_2 and player has hp not less than PERFECTION_2_2_CRITERIA
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get addAtk += PERFECTION_1_ADDATK if this perk has TIER1 2. Player will get addAtk += PERFECTION_2_1_ADDATK if this perk has TIER2_1 3. Player will get addAtk += PERFECTION_2_2_ADDATK if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk has been activated, deactivate this perk and 1. Player will get addAtk -= PERFECTION_1_ADDATK if this perk has TIER1 2. Player will get addAtk -= PERFECTION_2_1_ADDATK if this perk has TIER2_1 3. Player will get addAtk -= PERFECTION_2_2_ADDATK if this perk has TIER2_2
+ void setStartValues(Tier tier)	Do nothing

5.7.10. Class PrecisionStrikePerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase critical rate

Constructors

Name	Description
+ PrecisionStrike(Player player, Tier tier)	Constructor for PrecisionStrikePerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and 1. Player will get critRate += PRECISION_1_ADDCRITRATE if this perk has TIER1 2. Player will get critRate += PRECISION_2_1_ADDCRITRATE if this perk has TIER2_1 3. Player will get critRate += PRECISION_2_2_ADDCRITRATE but get critDmg -= PRECISION_2_2_MINUSCRITDAMAGE if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk has been activated, deactivate this perk and + Player will get critRate -= PRECISION_1_ADDCRITRATE if this perk has TIER1 + Player will get critRate -= PRECISION_2_1_ADDCRITRATE if this perk has TIER2_1 + Player will get critRate -= PRECISION_2_2_ADDCRITRATE but get critDmg += PRECISION_2_2_MINUSCRITDAMAGE if this per
+ void setStartValues(Tier tier)	Do nothing

5.7.11. Class RapidFirePerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase attack speed

Constructors

Name	Description
+ RapidFirePerk(Player player, Tier tier)	Constructor for RapidFirePerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and + Player will get addAttackSpeed += RAPID_NORMAL_ADDATKSPEED if this perk has TIER1 + Player will get addAttackSpeed += RAPID_2_1_ADDATKSPEED if this perk has TIER2_1 + Player will get addAttackSpeed += RAPID_NORMAL_ADDATKSPEED and get addMovementSpeed += RAPID_2_2_ADDMOVEMENTSPEED if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk has been activated, deactivate this perk and + Player will get addAttackSpeed -= RAPID_NORMAL_ADDATKSPEED if this perk has TIER1 + Player will get addAttackSpeed -= RAPID_2_1_ADDATKSPEED if this perk has TIER2_1 + Player will get addAttackSpeed -= RAPID_NORMAL_ADDATKSPEED and get addMovementSpeed -= RAPID_2_2_ADDMOVEMENTSPEED if this per
+ void setStartValues(Tier tier)	Do nothing

5.7.12. Class ReinforcedPerk extends BasePerk

- This class is used as a COMMON perk for players. It can increase movement speed

Constructors

Name	Description

+ ReinforcedPerk(Player player, Tier tier)	Constructor for ReinforcedPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If this perk hasn't been activated, activate this perk and + Player will get addDef += REINFORCED_1_ADDDEF if this perk has TIER1 + Player will get addDef += REINFORCED_2_1_ADDDEF if this perk has TIER2_1 + Player will get addDef += REINFORCED_2_2_ADDDEF but get addMovementSpeed -= REINFORCED_2_2_MINUSMOVEMENTSPEED if this perk has TIER2_2
+ void deactivatePerk(Player player)	If this perk has been activated, deactivate this perk and + Player will get addDef -= REINFORCED_1_ADDDEF if this perk has TIER1 + Player will get addDef -= REINFORCED_2_1_ADDDEF if this perk has TIER2_1 + Player will get addDef -= REINFORCED_2_2_ADDDEF but get addMovementSpeed += REINFORCED_2_2_MINUSMOVEMENTSPEED if this per
+ void setStartValues(Tier tier)	Do nothing

5.7.13. Class ThornPerk extends BasePerk

- This class is used as a COMMON perk for players. It can reflect damage from monsters

Constructors

Name	Description

+ ThornPerk(Player player, Tier tier)	Constructor for ThornPerk - Call super(player, tier, COMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	Do nothing
+ void deactivatePerk(Player player)	Do nothing
+ void setStartValues(Tier tier)	Do nothing
+ void slow(BaseMonster baseMonster)	If baseMonster hasn't been slowed, slow it by let addMovementSpeed -= THORN2_2_SLOW and after THORN_2_2_SLOWTIME let addMovementSpeed += THORN2_2_SLOW

5.7.14. Class TreasureHunterPerk extends BasePerk

- This class is used as a RARE perk for players. It can give a chance giving double rewards when opening a chest

Fields

Name	Description
- float chance	The chance of getting double reward
- boolean canRandom	Boolean to check whether player can get a chance to get double reward or not

Constructors

Name	Description
+ TreasureHunterPerk(Player player, Tier tier)	Constructor for TreasureHunterPerk - Call super(player, tier, RARE)

Methods

Name	Description

+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	If (canRandom) set player's isTreasureHunterActivate to randomChance() < chance and canRandom = false If this perk has TIER2_2, isMoreCoinDrop of each monster in the stage will be true
+ void deactivatePerk(Player player)	If this perk has TIER2_2, isMoreCoinDrop of each monster in the stage will be false
+ void setStartValues(Tier tier)	- If this perk has TIER2_1, chance will be TREASURE_2_1_REWARDCHANCE otherwise chance will be TREASURE_NORMAL_CHANCE - Set canRandom to false
+ void setCanRandomTreasure(boolean canRandom)	Setter for canRandom

5.7.15. Class UndeadPerk extends BasePerk

- This class is used as a UNCOMMON perk for players. When the player receives damage that is more than hp, player will survive with 1 hp

Fields

Name	Description
- boolean inCoolDown (= false)	The cooldown of this perk
- boolean hasStartTimer (= false)	Boolean to check whether the timer inside this perk is activated or not
- int activationDuration	The Activation Duration
- int cooldownDuration	The Cooldown Duration
- long startTime	The time when this perk start in each phase (Activating phase or Cooldown phase)
- SoundLoader UNDEAD_SOUND (= new	The sound when this perk is activated

SoundLoader(“UndeadActivate”, “sfx/sample_undead_activate.mp3”))	
- AnimationTimer deadTimer	Timer for activation
- AnimationTimer cooldownTimer	Timer for cooldown

Constructors

Name	Description
+ UndeadPerk(Player player, Tier tier)	Constructor for UndeadPerk - Call super(player, tier, UNCOMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return true
+ void activatePerk(Player player)	Do nothing
+ void deactivatePerk(Player player)	Do nothing
+ void setStartValues(Tier tier)	Do nothing
+ boolean isUndeadInCoolDown()	Getter for inCoolDown
- void createCooldownTimer()	Initialize cooldownTimer
- void createActivationTimer(Player player)	Initialize deadTimer
+ SoundLoader getUndeadSound()	Getter for UNDEAD_SOUND
+ void activateUndead(Player player)	if hasStartTimer is false, - Set activationDuration and cooldownDuration based on the tier - Set player’s currentHp to 1 - If this perk has TIER2_1, player’s addAtk += UNDEAD_2_1_ADDATK and addAttackSpeed += UNDEAD_2_1_ADDATKSPEED - activate deadTimer

	- Set hasStartTimer to true and UNDEAD_SOUND.play()

5.7.16. Class VampirismPerk extends BasePerk

- This class is used as a UNCOMMON perk for players. It can give a chance of restoring hp when dealing damage

Fields

Name	Description
- boolean canGainedHP	Boolean to check whether player can restore hp or not
- float vampirismChance	The chance of activating this perk

Constructors

Name	Description
+ VampirismPerk(Player player, Tier tier)	Constructor for VampirismPerk - Call super(player, tier, UNCOMMON)

Methods

Name	Description
+ boolean canUsePerk(Player player)	Return cainGainedHP && randomChance() < vampirismChance
+ void activatePerk(Player player)	- canGainedHP to false - If this perk has TIER2_1, player's currentHp is increased by VAMP_2_1_DRAINPERHIT percent of maxHp otherwise player's currentHp is increased by VAMP_NORMAL_DRAINPERHIT percent of maxHp
+ void deactivatePerk(Player player)	Set canGainedHP to false
+ void setStartValues(Tier tier)	- If tier ==TIER2_2, vampirismChance will be VAMP_2_2_DRAINCHANCE otherwise vampirismChance will be VAMP_NORMAL_CHANCE

	- setCanGainedHP(false)
+ void activateVampirism()	setCanGainedHP(true)
+ void setCanGainedHP(boolean canGainedHP)	Setter for canGainedHP

5.8. Package enums

5.8.1. Enum FaceDirection

- This enum represents entities' direction. It has some methods and contains the following values: LEFT, RIGHT

Methods

Name	Description
+ FaceDirection randomStartDirection()	return either LEFT or RIGHT. This is to random the first direction of a monster when that monster spawn
+ void switchDirection(BaseMonster baseMonster)	This is to switch monster's direction from LEFT to RIGHT and from RIGHT to LEFT

5.8.2. Enum Rarity

- This enum represents perks' rarity. It contains the following values: COMMON, UNCOMMON, RARE

5.8.3. Enum States

- This enum represents entities' state and objects' state. It has some methods and contains the following values: PLAYER_STAND_STILL, PLAYER_WALK, PLAYER_NORMAL_ATTACK, PLAYER_SPECIAL_ATTACK, PLAYER_DEAD, PLAYER_JUMP_UP, PLAYER_JUMP_DOWN, PLAYER_DASH, OBJECT, MONSTER_SPAWN, MONSTER_STAND_STILL, MONSTER_WALK, MONSTER_NORMAL_ATTACK, MONSTER_DEAD, BOSS_SKILL_ONE, BOSS_SKILL_TWO, BOSS_SKILL_THREE_STAND_STILL, BOSS_SKILL_THREE_WALK, BOSS_JUMP_UP, BOSS_JUMP_DOWN

5.8.4. Enum PerkTier

- This enum represents perks' tier. It contains the following values: TIER1, TIER2_1, TIER2_2

5.9. Package interfaces

5.9.1. Interface Attackable

- This interface defines method for BaseEntity showing that the entity can attack

Method

Name	Description
+ *boolean canAttack(BaseEntity baseEntity)*	This method is called to check whether the entity can attack or not
+ *void attack(Player player, BaseScene baseScene, int extraDamagePercent)*	This method is called when the entity is going to attack
+ *void isAttacked(BaseEntity baseEntity, BaseScene scene, int extraDamagePercent)*	This method is called when the entity is attacked
+ *hurt(BaseEntity attacker, BaseScene scene)*	This method is called when the entity loses his hp
+ *dead(BaseEntity attacker, BaseScene scene)*	This method is called when the entity is dead

5.9.2. Interface Moveable

- This interface defines a method showing that the BaseEntity and the BaseObject can move and it is triggered in MovingThread

Method

Name	Description
+ *void move(BaseScene scene)*	This method is called to move the entity/ the object

5.9.3. Interface Obtainable

- This interface defines a method showing that the BasePerk and the BaseReward can be collected by player

Method

Name	Description
+ *void collect(Player player)*	This method is called to let the player collect the perk/ the reward

5.9.4. Interface Perkable

- This interface defines methods showing that the BasePerk can activate its effect

Method

Name	Description
+ *boolean canUsePerk(Player player)*	This method is called in StatThread to check whether the perk can use its effect or not
+ *void activatePerk(Player player)*	This method is called in StatThread to activate the perk's effect
+ *void deactivatePerk(Player player)*	This method is called in StatThread to deactivate the perk's effect

5.9.5. Interface Tradable

- This interface defines methods showing that a player can buy BasePerk

Method

Name	Description
+ *boolean canBuy(Player player)*	This method is called to check whether a player can buy this perk or not
+ *void buy(Player player)*	This method is called to let a player buy this perk

5.9.6. Interface Upgradable

- This interface defines methods showing that BasePerk and player's class can be upgraded

Method

Name	Description
+ *boolean canUpgrade(Player player)*	This method is called to check whether the perk/ player's class can be upgraded or not
+ *void upgrade(Player player, Tier tier)*	This method is called to upgrade the perk/ player's class

5.10. Package objects

5.10.1. Abstract Class BaseObject implements Moveable

- This class is used as a base for all objects in the game

Fields

Name	Description
- Animation animation	Object's animation
- HitBox hitBox	Object's hitbox
- float speed	Object's move speed
- double posX	Object's position in axis X
- double posY	Object's position in axis Y
- FaceDirection faceDirection	Object's face direction
- String imgString	Object's image string
- int height	Object's height
- int width	Object's width
- ArrayList<String> FROM_PLAYER (= new ArrayList<>(Arrays.asList("NormalArrow", "ChargeArrow")))	Arraylist that contain object's type that come from player
- boolean willRemove (= false)	Boolean for checking whether that will remove
- boolean canIgnoreTile	Boolean for checking whether that can ignore tile
- boolean canFall	Boolean for checking whether that can fall

- boolean isDamageDealt	Boolean for checking whether that damage has already been dealt
- boolean canInteract	Boolean for checking whether that player can interact with
- double distance	Distance travelled from owner
- BaseEntity owner	Entity that create this object
- ArrayList<String> GIF_CLASS (= new ArrayList(Arrays.asList("HPDropSmall", "HPDropLarge", "BowSymbol", "DaggerSymbol", "CoinReward", "HPReward")))	Arraylist that contain object's type that need to read img with .gif
- Group popUp() (= new Group())	Group of popup UI
- ArrayList<SoundLoader> soundList	Arraylist that contain soundList

Constructors

Name	Description
+ BaseObject(BaseEntity owner, double posX, double posY, FaceDirection faceDirection)	Constructor for BaseObject - Set position in axis X to posX - Set position in axis Y to posY - Set face direction to faceDirection - Set speed to NOT_MOVE - Set canIgnoreTile to false - Set canFall to false - Set damageDealth to false - Set canInteract to false - Set the image string based on whether it's a GIF_CLASS or not - Set owner to owner

Methods

Name	Description
+ void move(BaseScene scene)	Move object - If distance more than owner's attack range and is not ThrowerBomb, setWillRemove to true - If can't ignore tile, call

	checkHitTile(); otherwise, call moveObject()
	- If from player, call checkHitMonster(); otherwise, call checkHitPlayer()
	- If can fall
	  - If not jump, call checkFall()
	  - If jump, call jump()

- void checkHitTile(double moveSpeed, BaseScene scene)	Check if object hit tile when move - If face direction is LEFT, multiply movespeed with -1 - If object can move, move object position; otherwise, if object is ThrowerBomb, activated bomb; otherwise, setWillRemove to true
- void moveObject(double moveSpeed)	Move object - Move object position

- void checkHitMonster(BaseScene scene)	Check if object hit monster
	- Check for every monster if object is hit monster
	  - If object is NormalArrow and monster can take damage, attack monster and setWillRemove to true
	  - If object is ChargeArrow and monster can take damage, attack monster
	  - If object is StompEffect that come from player and monster can take damage, attack monster

- void checkHitPlayer(BaseScene scene)	Check if object hit player
	- If object hit player and player can take damage, call makeDamage()
	  - If object is HPDropSmall or HPDropLarge, call heal()
	  - If object is Shroom, call superJump()
	  - If object can interact, call moveInInteractObject()
	- If object is not hit player and object can interact, call moveOutInteractObject()

- void makeDamage(Player player, BaseScene scene)	Object make damage to player - If object is Missile, player get attacked and setWillRemove to true; otherwise - If object not dealt damage yet, make damage player with extra damage for some object's type
- void makeThrowerBombDamage(Player player, BaseScene scene)	ThrowerBomb make damage to player - If the bomb is already activated, damage to the player and setDamageDealt to true. If bomb is from IceThrowerGiGee call applyIceThrowerSlow() - If the bomb does not activate, activate.
- void applyIceThrowerSlow(Player player)	Apply slow to player - Calculate the movementSlowAmount and dashSlowAmount to be not more than the maximum, and then apply them to the player - Call setSlowTimer()
- void setSlowTimer(Player player, int movementSlowAmount, int dashSlowAmount)	Duration of slow effect - Make AnimatimerTimer to make slow effect apply for ICETHROWERGIGEE_SLOW_DURATION
- void heal(Player player)	Heal player - Increase player's current health, and setWillRemove to true and play get heal sound
- void activateSuperJump(Player player, BaseScene scene)	Activate super jump when land on Shroom - Check if player in state JUMP_DOWN - setyVelocity to player and run animation to Shroom
- void moveInsideInteractObject(BaseScene scene)	Check if player move inside object that can interact - Set canInteract to true - Set object that player interact with to this object - If object is merchant or symbol,

	show popup
- void moveOutInteractObject(BaseScene scene)	Check if player move out object that can interact - Set canInteract to false - Set object that player interact with to null - If object is merchant or symbol, remove popup
- void checkFall(BaseScene scene)	Check if object will fall - If object can move 1 pixel below, setJump to true and setyVelocity to 0
- void jump(BaseScene scene)	Object jump - If the object can ignore tiles, move the object; otherwise, check if it can move. If it can move, then move. If it can't move and it is a ThrowerBomb, call activateBomb()
- void createMerchantPopUp(Merchant merchant, BaseScene scene)	Create merchant popup - call createPotionPrice() - Add popup to scene
- void removePopUp(BaseScene scene)	Remove popup - Remove popup
- void addClassSymbolPopUp(Symbol symbol, BaseScene scene)	Add class symbol popup - Call createClassDetail() - Add popup to scene
- void createClassDetail(Symbol symbol)	Create class detail popup - Create class popup imageView and add to scene
- void addPerkSymbolPopUp(BasePerk perk, BaseScene scene)	Create perk popup - Call createPerkDetail() - If this scene is MarketScene, call createPerkPrice() and add to scene
- void createPotionPrice(Merchant merchant)	Create potion’s price - Create potion’s price and add to scene
- void createPerkPrice(BasePerk perk)	Create perk’s price - Create perk’s price and add to scene

- void createPerkDetail(BasePerk perk)	Create perk's detail - Create perk's detail and add to scene
- boolean isHitEntity(BaseEntity baseEntity)	Check if object hit entity - Call inEntityHitBox() at multiple position
- boolean checkEntityHitBox(double poX, BaseEntity baseEntity)	Check if object in entity's hitbox - Check if position x in entity's hitbox
- void runOneTimeAnimation(int frames, int framesPerSec, BaseScene scene, boolean willRemove)	Run animation one time - Create AnimationTimer to run animation of object one time and then remove from scene and player if willRemove is true
+ Animation getAnimation()	Getter of animation
+ void setAnimation(Animation animation)	Setter of animation
+ float getSpeed()	Getter of speed
+ void setSpeed(float speed)	Setter of speed
+ double getPosX()	Getter of posX
+ void setPosX(double posX)	Setter of posX
+ double getPosY()	Getter of posY
+ void setPosY(double posY)	Setter of posY
+ FaceDirection getFaceDirection()	Getter of faceDirection
+ void setFaceDirection(FaceDirection faceDirection)	Setter of faceDirection
+ String getImgString()	Getter of imgString
+ void setImgString(String imgString)	Setter of imgString
+ HitBox getHitBox()	Getter of hitBox
+ void setHitBox(HitBox hitBox)	Setter of hitBox
+ int getHeight()	Getter of height
+ void setHeight(in height)	Setter of height

+ int getWidth()	Getter of width
+ void setWidth(int width)	Setter of width
+ boolean willRemove()	Getter of willRemove
+ void setWillRemove(boolean willRemove)	Setter of willRemove
+ double getDistance()	Getter of distance
+ void setDistance(double distance)	Setter of distance
+ BaseEntity getOwner()	Getter of owner
+ void setOwner(BaseEntity owner)	Setter of owner
+ Group getPopUp()	Getter of popUp
+ ArrayList<SoundLoader> getSoundList()	Getter of soundList
+ void setSoundList(ArrayList<SoundLoader> soundList)	Setter of soundList
+ void setCanIgnoreTile(boolean canIgnoreTile)	Setter of canIgnoreTile
+ void setCanFall(boolean canFall)	Setter of canFall
+ boolean isDamageDealt()	Getter of isDamageDealt
+ void setDamageDealt(boolean damageDealt)	Setter of isDamageDealt
+ boolean canInteract()	Getter for canInteract
+ void setCanInteract(boolean canInteract)	Setter for canInteract

5.10.2. Abstract Class BaseReward extends BaseObject implements Obtainable

- This class is used as a base for rewards that player can obtain

Constructors

Name	Description
+ BaseReward(BaseEntity owner,	Constructor for BaseReward

double posX, double posY, FaceDirection faceDirection)	- Call super()

5.10.3. Class BlackSmith extends BaseObject

- This class is used as blacksmith that upgrade perk in market scene

Fields

Name	Description
- String type	Type of blacksmith
- boolean isStop	For stop run animation

Constructors

Name	Description
+ BlackSmith(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, String type)	Constructor for BlackSmith - Call super() - setType to type - setWidth, setHeight - Set canIgnoreTile and canInteract to true - Set image string - Set animation and set viewport to starter frame - Set hitbox - Call runAnimation()

Methods

Name	Description
- void runAnimation(int typeNumber)	Make AnimationTimer to run animation of blacksmith
- disappear(BaseScene scene)	Make blacksmith disappear from scene
- String getType()	Getter for type
- void setType(String type)	Setter for type
- boolean isStop()	Getter for isStop
- void setStop(boolean stop)	Setter for isStop

5.10.4. Class Bomb extends BaseObject

- This class is used as the bomb effect when KleeGiGee dies and BigBoy dies

Fields

Name	Description
- String type	Type of bomb

Constructors

Name	Description
+ Bomb(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, BaseScene scene)	Constructor for Bomb - Call super() - setWidth, setHeight, setType based on owner - Set canIgnoreTile to true - Set image string - Set animation and set viewport to starter frame - Set hitbox - Set soundList - Call runOneTimeAnimation()

5.10.5. Class ChargeArrow extends BaseObject

- This class is used as an arrow of Bow Class and will be launched when Bow user uses special attack

Fields

Name	Description
- int damageIncreasing	Amount of damage increasing

Constructors

Name	Description
+ ChargeArrow(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, int damageIncreasing)	Constructor for ChargeArrow - Call super() - setWidth, setHeight, setSpeed - Set animation and hitbox - Set damage increasing to damageIncreasing

Methods

Name	Description
+ int getDamageIncreasing()	Getter for damageIncreasing
+ void setDamageIncreasing(int	Setter for damageIncreasing

damageIncreasing)	

5.10.6. Class CoinReward extends BaseReward

- This class is a type of reward that gives coin to the player

Constructors

Name	Description
+ CoinReward(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for CoinReward - Call super() - setWidth, setHeight - Set canIgnoreTile and canInteract to true - Set animation and hitbox

Methods

Name	Description
+ void collect(Player player)	Player collect coin reward - Increase player's coin

5.10.7. Class Door extends BaseObject

- This class is used as a door and when player interacts, player will go to the next stage

Fields

Name	Description
- String doorType	Type of door
- boolean open (= false)	Boolean to checking whether this door is open

Constructors

Name	Description
+ Door(BaseEntity owner, double posX, double posY, FaceDirection faceDirection, String type)	Constructor for Door - Call super() - Set doorType to type - setWidth, setHeight - Set canIgnoreTile and canInteract to true - Set image string - Set animation and hitbox

Methods

Name	Description
+ String getDoorType()	Getter for doorType
+ void openDoor()	Open door - If door not broken, change image string, set new animation - setOpen to true
+ boolean isOpen()	Getter for isOpen
+ void setOpen(boolean open)	Setter for isOpen

5.10.8. Class Fire extends BaseObject

- This class is used as a fire for BigBoy attack

Constructors

Name	Description
+ Fire(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, BaseScene scene)	Constructor for Fire - Call super() - setWidth, setHeight - Set canIgnoreTile to true - Set animation and set viewport to first frame and set hit box - Call runRepeatAnimaion()

Methods

Name	Description
- void runRepeatAnimation(BaseScene scene)	Create AnimationTimer to run animation repeatedly
- void stopFire(BaseScene scene, BaseObject fire)	Stop fire animation - Remove fire animation from scene - Set attack delay to 2 - Play BossFire sound - Call createBossCooldownSkill3()
- void createBossCooldownSkill3(Base Boss boss)	Create animation timer to cooldown boss's skill 3

5.10.9. Class HPDropLarge extends BaseObject

- This class is used as a health potion that can restore much hp to player

Constructors

Name	Description
+ HPDropLarge(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for HPDropLarge - Call super() - setWidth, setHeight - Set canFall to true - Set animation and hitbox

5.10.10. Class HPDropSmall extends BaseObject

- This class is used as a health potion that can restore some hp to player

Constructors

Name	Description
+ HPDropSmall(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for HPDropSmall - Call super() - setWidth, setHeight - Set canFall to true - Set animation and hitbox

5.10.11. Class HPReward extends BaseReward

- This class is a type of reward that increases player's hp capacity and also restores hp

Constructors

Name	Description
+ HPReward(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for HPReward - Call super() - setWidth, setHeight - Set canIgnoreTile and canInteract to true - Set animation and hitbox

Methods

Name	Description
+ void collect(Player player)	Player collect HP reward - Call calculateNewHp()

5.10.12. Class Merchant extends BaseObject implement Tradable

- This class is used as a merchant that sell potion in market scene

Constructors

Name	Description
+ Merchant(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for Merchant - Call super() and generateNewPrice() - setWidth, setHeight - Set canIgnoreTile and canInteract to true - Set animation and set viewport to first frame and set hitbox

Methods

Name	Description
+ boolean canSell(Player player)	Return true if player's coin more than price, player's health not full and merchant still sell
+ void sell(Player player, BaseSence scene)	Sell potion - Reduce player's coin - Change viewport of merchant to make potion disappear - Check discountMasterPerk - setSold to true - Call generateNewPrice()
+ int calculateDiscountPrice(Player player)	Check discount from discountMasterPerk and apply to perk
- void generateNewPrice()	Random price between MIN_POTION_PRICE and MAX_POTION_PRICE
- void removePotion(Player player, BaseScene scene)	Set canInteract to false Set object that interact with to null Remove popup and hitbox from scene and remove from player
+ boolean isSold()	Getter for isSold
+ void setSold(Boolean isSold)	Setter for isSold
+ int getPrice()	Getter for price
+ void setPrice(int price)	Setter for price

5.10.13. Class Missile extends BaseObject

- This class is used as a missile for ArcherGiGee attack

Constructors

Name	Description
+ Missile(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for Missile - Call super() - setWidth, setHeight, setSpeed - Set animation and hitbox

5.10.14. Class NormalArrow extends BaseObject

- This class is used as an arrow of Bow Class and will be launched when Bow user uses normal attack

Constructors

Name	Description
+ NormalArrow(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for NormalArrow - Call super() - setWidth, setHeight, setSpeed - Set animation and hitbox - Set distance to width

5.10.15. Class PerkChestReward extends BaseObject

- This class is used as a chest storing perks for player

Fields

Name	Description
- int startX (= 4)	Start position to read animation in x-axis
- int startY (= 4)	Start position to read animation in y-axis
- boolean isOpen (= false)	Boolean to checking whether this chest is open

Constructors

Name	Description
+ PerkChestReward(BaseEntity baseEntity, double posX, double posY, FaceDirection	Constructor for PerkChestReward - Call super() - setWidth, setHeight

faceDirection)	- Set canIgnoreTile and canInteract to true - setAnimation and set viewport to first frames and set hitbox

Methods

Name	Description
+ void openChest()	Open chest - Play PerkChestOpen sound - Change viewport - setOpen to true
+ boolean isOpen()	Getter for isOpen
+ void setOpen(boolean open)	Setter for isOpen

5.10.16. Class ShockWave extends BaseObject

- This class is used as a shockwave for BigBoy attack

Constructors

Name	Description
+ ShockWave(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, BaseScene scene)	Constructor for ShockWave - Call super() - setWidth, setHeight - Set canIgnoreTile to true - Set animation and set hitbox - Call runOneTimeAnimation()

5.10.15. Class Shroom extends BaseObject

- This class is used as a shroom in the game and when player jumps on it, player will be bounced

Constructors

Name	Description
+ Shroom(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for Shroom - Call super() - setWidth, setHeight - Set canIgnoreTile to true - Set animation and set hitbox

5.10.18. Class StompEffect extends BaseObject

- This class is used as a stomp effect for BigBoy attack

Constructors

Name	Description
+ StompEffect(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, BaseScene scene)	Constructor for StompEffect - Call super() - setWidth, setHeight - Set canIgnoreTile to true - Set animation and set hitbox - Set soundList - Call runOneTimeAnimation()

5.10.19. Class Symbol extends BaseObject

- This class is used as a symbol and when player interacts, it will show popup about its data

Fields

Name	Description
- ArrayList<String> CLASSLIST (= new ArrayList<>(Arrays.asList("Bow", "Broadsword", "Dagger")))	ArrayList that contain class's type
- String symbolType	Symbol's type
- String popUpString	Popup image string
- BasePerk perk (= null)	Perk

Constructors

Name	Description
+ Symbol(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, String type, BasePerk perk)	Constructor for Symbol - Call super() - setWidth, setHeight - Set canIgnoreTile and canInteract to true - setPerk to perk - Set image string and popup string based on type - Set animation and set hitbox

Methods

Name	Description
+ String getSymbolType()	Getter for symbolType
+ void setSymbolType(String symbolType)	Setter for symbolType
+ BasePerk getPerk()	Getter of perk
+ void setPerk(BasePerk perk)	Setter for perk
+ String getPopUpString()	Getter of popUpString
+ void setPopUpString(String popUpString)	Setter for popUpString

5.10.20. Class ThrowerBomb extends BaseObject

- This class is used as a bomb for ThrowerGiGee attack

Fields

Name	Description
- boolean isActivate (= false)	Boolean to checking whether this is already activate
- String throwerType	Thrower's type

Constructors

Name	Description
+ ThrowerBomb(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection, double speed, double yVelocity, String type)	Constructor for ThrowerBomb - Call super() - setWidth, setHeight, setSpeed - Set canIgnoreTile and canFall to true - setThrowerType based on type - Set image string and popup string based on type - Set animation and set hitbox - setJump to true - setyVelocity to yVelocity - Call canIgnoreTileTimer()

Methods

Name	Description
- void canIgnoreTileTimer(String	Create AnimationTimer to count

type)	cooldown for changing canIgnoreTile state
+ void activateBomb(BaseScene scene)	Activate bomb - setSpeed to NOT_MOVE - Set canIgnoreTile, canFall and activate to true - If type is “Default”, call setDefaultSetUp - If type is “Ice”, call setIceSetUp
- void setDefaultSetUp(BaseScene scene)	Setup based on type - setWidth, setHeight, setImage, set position and set hitbox - Call runOneTimeAnimation()
- void setIceSetUp(BaseScene scene)	Setup based on type + setWidth, setHeight, setImage, set position and set hitbox + Call runOneTimeAnimation()
+ boolean isActivate()	Getter for isActivate
+ void setActivate(boolean activate)	Setter for isActivate
+ String getThrowerType()	Getter for throwerType
+ void setThrowerType(String throwerType)	Setter for throwerType

5.10.21. Class VendingMachine extends BaseObject

- This class is used as a vending machine in ClassSelectionScene

Fields

Name	Description
- boolean isStop (= false)	Use to stop animation

Constructors

Name	Description
+ VendingMachine(BaseEntity baseEntity, double posX, double posY, FaceDirection faceDirection)	Constructor for VendingMachine - Call super() - setWidth, setHeight - Set canIgnoreTile and canInteract to true

	- Set animation and set viewport to first frames and set hitbox - setSoundList - Call runAnimation()

Methods

Name	Description
- void runAnimation()	Create AnimationTimer to run animation
+ boolean isStop()	Getter for isStop
+ void setStop(boolean isStop)	Setter for isStop

5.11. Package scenes

5.11.1. Abstract Class BaseScene extends Pane

- This class is used as a base for most scenes in the game

Fields

Name	Description
- CanvasDrawer bgDrawer (= null)	Background of scenes
- CanvasDrawer minimapDrawer (= null)	Draw minimap
- CanvasDrawer entitiesInMinimap (= null)	All entities in map show in minimap
- Image bgImg	Background image
- ImageView bgImgView	Background imageView
- ImageView healthUI	Health UI of player
- Rectangle healthBar	Health bar of player
- Text healthText	Text that show player current hp
- ImageView bossHealthBarFrame	Frame of boss health bar frame
- Rectangle bossHealthBar	Health bar of boss
- ImageView bossHealthBarBg	Background of boss health bar
- ImageView coinUI	Coin that show in game
- Text coinText	Amount of coin that player have

- ImageView remainMonsterUI	Icon remaining monster in the scene
- Text remainMonsterText	Amount of monsters in scene
- ImageView dashIcon (= new ImageView())	Dash icon
- ImageView firstIconFrame	Frame of dash icon
- ImageView secondIconFrame	Frame of special attack icon
- Text dashRemaining (= new Text())	Amount of dash left
- Rectangle dashCoolDown (= new Rectangle())	Rectangle that show cooldown of dash
- ImageView specialAttackIcon (= new ImageView())	Special attack icon
- Rectangle specialAttackCoolDown (= new Rectangle())	Rectangle that show cooldown of special attack
- ImageView minimapBgUI	Background of minimap
- Group playerUI (= new Group())	Group of player UI
- Group inventoryUI (= new Group())	Group of inventory UI
- Group perkChestUI	Group of chest UI
- int[][] lvlData	Level data of a scene, used to determine where characters can land and walk
- Player player	Player in game
- Set<KeyCode> pressedKeys (= new HashSet<>())	Set that contain what key player press
- Stage stage	Stage
# PlayerAnimationThread playerAnimationThread	Thread that run player animation
# MovingThread movingThread	Thread that run moving
# MonsterAnimationThread monsterAnimationThread	Thread that run monster animation thread
# BotLogicThread botLogicThread	Thread that run bot logic

# StatThread statThread	Thread that run perk and status
- ArrayList<Door> doors (= new ArrayList<>())	ArrayList that contain door in map
- boolean isPause (= false)	Boolean to checking whether that is pause

Constructors

Name	Description
+ BaseScene(Stage stage)	Constructor for BaseScene - Set stage to stage - Set fullscreen to true

Methods

Name	Description
+ void drawUI()	Call createHpGroup(), createSkillIcon, createMiniMap(), createStageDetails(), createBossHealthBar(), addElementsToPlayerUI() and set position for playerUI and add to scene
- void createHpGroup()	Calculate current hp and maxHp and position
- void createSkillIcon()	Create imageView of icon frame and dash, special skill icon by calling createFirstIcon(), create secondIcon(), create dashIcon(), create specialAttackIcon()
- void createFirstIcon(Image iconFrameImg)	Set position of first icon frame
- void createSecondIcon(Image iconFrameImg)	Set position of second icon frame
- void createDashIcon(Image dashIconImg)	Set position of dash icon
- void createSpecialAttackIcon(Image specialAttackIconImg)	Set position of special attack icon
- void createMinimap()	Create minimap based on size of background

- void createStageDetails()	Call createCoinGroup(), createMonsterRemainingGroup(), creatWaveNumberGroup()
- void createBossHealthBar()	Create bossHealthBarFrame, bossHealthBarBg and boss HealthBar and set position to all of them
- void createCoinGroup(String currentCoin)	Create coin icon and amount of coins in screen
- void adjustCoinGroup(Image coinImg, double extraWidth, double extraHeight, double UI_X, double UI_Y, double textX, double textY)	Set position of coin icon and amount of coins
- void createMonsterRemainingGroup(String currentCoin)	Create remainMonsterUI and remainMonsterText and set position to all of them
- void createWaveNumberGroup(String remainMonster)	Create wave text and set position
- void addElementsToPlayerUI()	Add all of UI to playerUI
+ void updateUI()	Call updateHpGroup(), updateStageDetails(), updateMiniMap(), updateBossHealthBar() and if game is not pause set player and playerUI to front
- void updateHpGroup()	Update hp group to match with current hp
- void updateBossHealthBar()	Update hp of bar to match with boss's current hp
- void updateStageDetails()	Call updateCoin(), updateMonsterRemaining(), updateDash(), updateWaveNumber()
- void updateCoin(String currentCoin)	Set text to match current amount of coins
- void updateMonsterRemaining(String currentCoin)	Set position and amount of monsters
- void updateDash()	Update text to match current dash remaining

- void updateWaveNumber(String remainMonster)	Update wavenumber to match current wave and if clear all waves will show “Complete”
+ void updateMinimap()	Update entities position in minimap to match entities position in real game
+ void drawBackground()	Draw background, minimap and entities in map
+ void createInventoryUI()	Show inventory UI that has player’s stat and perk that player obtain
- void drawAllPerks()	Call drawperk()
- void drawPerk(BasePerk perk, float posX, float posY)	Call createPerkIcon() and createPerkText()
- void createPerkText(BasePerk perk, float posX, float posY)	Create text that shown tier of perk
- void createPerkIcon(BasePerk perk, float posX, float posY)	Create icon of perk
- Imageview showMiniPopUp(BasePerk perk, double posX, double posY)	Return popup imageView that will show when player hover mouse over perk icon
- void pickPerkToUpgrade(BasePerk perk)	Call createPerkChoiceToUpgrade() and add it to upgradeUI
- void createPerkChoiceToUpgrade(BasePerk perk, Tier tier, double posX, double posY)	Call setUpChoice() and setUpPriceGroup() and all in choiceUI
- setUpChoice(ImageView choice, double posX, double posY, BasePerk perk, Tier tier)	Set position of upgrade choice and set on mouse click to choose choice
- void setUpPriceGroup(ImageView choice, BasePerk perk, Group priceGroup)	Set position of coinUI and coinText to show coin that need to use for upgrade perk
- Group createMenuButton()	Create menu button at right bottom side of the screen that can click to back to main menu
- void addAllStats(double scaleForInventory, ImageView	Add all stat to inventoryUI that match will player’s stat

inventoryBg, Group button)	
+ void createUpgradeUI()	Create ui when interact with blacksmith that show perk
- Group createBackButton()	Create back button at right bottom side of the screen that can click to close upgradeUI
+ void createPerkChestUI()	Create UI when interact with perkChest
- void setPerkChestBackground(ImageView perkChestBg)	Set position of perkChestBackground
- void addSelectedPerksForChest(ArrayList<BasePerk> perks, ImageView perkChestBg, BaseScene scene)	Set position of perk to perkChestUI
- void selectPerk(BasePerk perk, BaseScene scene)	Add perk to player and close perkChestUI
+ void setKey(Animation animation)	Set key - setOnKeyPressed(), setOnKeyRelease(), setOnMouseClick(), setOnMousePressed(), setOnMouseRelease()
- void handleKeyPressed(Animation animation)	Set key to each key - TAB : pressInventory() - A : pressMoveLeft() - D : pressMoveRight() - E : interact() - SHIFT : activateDash() - SPACE : activateJump() - SPACE + S : activatePassingTheFloor()
- void interact()	If player can interact, call interact based on scene
- void interactInClassSelectionScene()	If player interact with symbol, call interactSymBol() If player interact with vending machine, call createSymbol() and set vending machine to can't interact
- void interactInFightScene()	If player interact with baseReward, call

	obtainBaseReward() If player interact with perkChestReward, call obtainPerkChestReward() If player interact with door, call nextStage() If player interact with blacksmith call createOpenDoor() and createUpgradeUI()
- void interactInMarketScene()	If player interact with symbol, call pickPerkToBuy() If player interact with merchant and merchant can sell, call merchant.sell() If player interact with blacksmith, call createUpgradeUI()
- void pickPerkToBuy(Symbol perkSymbol)	Buy perk and remove call removePerk()
- void removePerk(Symbol perkSymbol)	Remove perk animation, hitbox, popup in scene
- void activatePassingTheFloor()	Move player down 2 pixel if below player is passableTile
- void activateJump(Animation animation)	Set jump to true and set Velocity to JUMP_VELOCTIY
- void activateDash(Animation animation)	Set state to PLAYER_DASH and set dash to true
- void obtainBaseReward()	Collect reward and open door
- void obtainPerkChestReward()	Create open door and call createPerkChestUI()
- void pressInventory()	If pause, remove inventoryUI and start thread If not pause, call stopAllThread() and createInventoryUI()
- void nextStage()	Stop all thread, reset fight scene and start new fight scene based on door type that player interact with
- void interactSymbol(Symbol symbol)	Select class to symbol's class and set vending machine back to can interact and remove perk on ground
- void pressMoveLeft(Animation animation)	setMoveLeft to true if animation not move right

- void pressMoveRight(Animation animation)	setMoveRight to true if animation not move left
- void removeClosedDoor()	Remove door animation and call door.openDoor()
- void selectClass(String className)	Reset previous player and call setPlayerClass() to make new player base on symbol player interact
- void setPlayerClass(String className)	setPlayer to new class based on symbol player interact
- void setPlayerPosition(String className)	Set new player position to same position of previous player
- void setFaceDirection(FaceDirection previousFaceDirection)	setScaleX() based on face direction
+ void startAllThreads(BaseScene baseScene)	Start all thread - Start playerAnimationThread - Start monsterAnimationThread - Start movingThread - Start statThread - Start botLogicThread
- void handleKeyReleased(KeyCode keyCode, Animation animation)	Set on key release - A : releaseMoveLeft() - D : releaseMoveRight()
- void releaseMoveRight(Animation animation)	Set player back to stand if not move left
- void releaseMoveLeft(Animation animation)	Set player bak to stand if not move right
- void handleMouseClicked(MouseButton button, Animation animation)	Set on mouse click - PRIMARY : activateNormalAttack() - SECONDARY : activateSpecialAttack()
- void activateNormalAttack(Animation animation)	Set state to player normal attack and disable movement
- void activateSpecialAttack(Animation animation)	Activate special skill based on player's class

- void handleMousePressed(MouseButton button, Animation animation)	Set on mouse press - activateBowSpecialAttack() if player's class is Bow and not in cooldown
- void activateBowSpecialAttack(Animation animation)	Set state to PLAYER_SPECIAL_ATTACK and set hold RMB to true
- void activateBroadSwordSpecialAttack(Animation animation)	Set state to PLAYER_SPECIAL_ATTACK
- void activateDaggerSpecialAttack(Animation animation)	Set state to PLAYER_SPECIAL_ATTACK if can warp to nearest monster
- void handleMouseReleased(MouseButton button, Animation animation)	Set on mouse release - SECONDARY : if isHoldRMB() is true, set hold RMB to false
- void blink(ImageView imageView)	Blink - Create AnimationTimer to adjust brightness to animation and make it blink
+ void setScenePosition()	Set position when spawn and warp to not over a background image
+ String toString()	Return getClass().setSimpleName()
+ void stopAllThreads()	Stop all thread - Stop playerAnimationThread - Stop movingThread - Stop monsterAnimationThread - Stop botLogicThread - Stop statThread
+ void setLowBrightness()	Set brightness of animation to -0.5
+ void setDefaultBrightness()	Set brightness of animation to 0
+ Player getPlayer()	Getter of player
+ void setPlayer(Player player)	Setter of player
+ Image getBgImg()	Getter of bgImg
+ void setBgImg(Image bgImg)	Setter o bgImg
+ ImageView getBgImgView()	Getter of bgImgView

+ void setBgImgView(ImageView imageView)	Setter of bgImgView
+ Set<KeyCode> getPressedKeys()	Getter of pressedKeys
+ int[][] getLevelData()	Getter of lvlData
+ void setLevelData(int[][] lvlData)	Setter of lvlData
+ Stage getStage()	Getter of stage
+ Group getPlayerUI()	Getter of playerUI
+ ArrayList<Door> getDoors()	Getter of doors
+ void setDoors(ArrayList<Door> doors)	Setter of doors
+ boolean isPause()	Getter of isPause
+ void setPause(boolean pause)	Setter of isPause
+ ImageView getDashIcon()	Getter of dashIcon
+ Rectangle getDashCoolDown()	Getter of dashCoolDown
+ ImageView getSpecialAttackIcon()	Getter of specialAttackIcon
+ Rectangle getSpecialAttackCoolDown()	Getter of specialAttackCoolDown

5.11.2. Class BossScene extends BaseScene

- This class is used as a scene when player fights against boss

Fields

Name	Description
- BossScene instance	BossScene itself

Constructors

Name	Description
+ BossScene(Stage stage, Player player)	Constructor for BossScene - Call super() - Clear player object - Set player - Setkey to animation

	- Set level data to bossSceneData - Set wave number to 0 - Set background - Call drawBackground() - Spawn player - Start all thread - Call drawUI(), stop all background song - Play bossSceneSong

Methods

Name	Description
+ BossScene getBossScene(Stage stage, Player player)	If instance not null return instance If instance is null, create new bossScene
+ void resetBossScene()	Set instance to null

5.11.3. Class ClassSelectionScene extends BaseScene

- This class is used as a scene for choosing a class

Fields

Name	Description
- ClassSelectionScene instance	ClassSelectionScene itself

Constructors

Name	Description
+ ClassSelectionScene(Stage stage)	Constructor for ClassSelectionScene - Call super() - Set player - Setkey to animation - Set level data to classSelectionSceneData - Set background - Spawn player - Start all thread - Call createAllClassSymbols() and createVendingMachine() - Call drawUI(), stop all background song - Play classSelectionScene song

Methods

Name	Description
- void createAllClassSymbols()	Create all symbol and add to scene
- void createVendingMachine()	Create vending machine and add to scene
+ ClassSelectionScene getClassSelectionScene(Stage stage)	If instance not null return instance If instance is null, create new classSelectionScene
+ void resetClassSelectionScene()	Set instance to null

5.11.4. Class DeathScene extends GridPane

- This class is used as a scene when player dies

Fields

Name	Description
- DeathScene instance	DeathScene itself

Constructors

Name	Description
+ DeathScene(Stage stage)	Constructor for DeathScene - Set background to black - Set padding to 10 - Set alignment to CENTER - createColumns and createRows - Create deathText - Create restartButton - Create exitButton

Methods

Name	Description
+ DeathScene getDeathScene(Stage stage)	If instance not null return instance If instance is null, create new deadScene

5.11.5. Class FightScene extends BaseScene

- This class is used as a scene when player fights against GiGees

Fields

Name	Description

- FightScene instance	FightScene itself
- int numberOfFightScene (= 0)	Number of fight scene that player play
- String roomType	Type of reward in room

Constructors

Name	Description
+ FightScene(Stage stage, Player player, String roomType)	Constructor for FightScene - Call super() - Clear player object - Set player - Setkey to animation - Set level data to random scene in all fight scenes - Set wave number to 0 - Set background - Call drawBackground() - Spawn player - Start all thread - Call drawUI(), stop all background song - Play random fightSceneSong

Methods

Name	Description
- SoundLoader getRandomFightSceneSong()	Random fight scene song and return song
+ FightScene getFightScene(Stage stage, Player player, String roomType)	If instance not null return instance If instance is null, create new fightScene
+ void resetFightScene()	Set instance to null
+ String getRoomType()	Getter of roomType
+ void setRoomType(String roomType)	Setter of roomType
+ int getNumberOfFightScene()	Getter of numberOfFightScene
+ void setFightSceneSong(SoundLoader fightSceneSong)	Setter of fightSceneSong

5.11.6. Class MarketScene extends BaseScene

- This class is used as a scene that let player buy or upgrade something

Fields

Name	Description
- MarketScene instance	MarketScene itself

Constructors

Name	Description
+ MarketScene(Stage stage, Player player)	Constructor for MarketScene - Call super() - Clear player object - Set player - Setkey to animation - Set level data to marketSceneData - Set background - Call drawBackground() - Spawn player - Start all thread - Call createAllPerkSymbols(), createMerchant(), createBlackSmith() - Call drawUI(), stop all background song - Play marketScene song

Methods

Name	Description
+ void createAllPerkSymbols(Player player)	Call createSymbol at specific position
- void createMerchant()	Create new merchant and add to scene
- void createBlackSmith()	Create new blackSmith and add to scene
+ MarketScene getMarketScene(Stage stage, Player player)	If instance not null return instance If instance is null, create new marketScene
+ void resetMarketScene()	Set instance to null

5.11.7. Class OptionScene extends GridPane

- This class is used as a scene to adjust the sound volume

Fields

Name	Description
- OptionScene instance	OptionScene itself
- Slider soundSlider (= new Slider(0, 1, 0.5))	Slider to adjust sound volume

Constructors

Name	Description
+ OptionScene(Stage stage)	Constructor for OptionScene - Set background to black - Set padding to 10 - Set alignment to CENTER - createColumn and createRows - Create optionText - Create soundSlider

Methods

Name	Description
+ OptionScene getOptionScene(Stage stage)	If instance not null return instance If instance is null, create new optionScene

5.11.8. Class StartScene extends GridPane

- This class is used as a start scene

Fields

Name	Description
- StartScene instance	StartScene itself

Constructors

Name	Description
+ StartScene(Stage stage)	Constructor for StartScene - Set background to white - Set padding to 10 - Set alignment CENTER - createColumns and createRows - Create nameLabel

	- Create start button - Create option button - Create exit button - Set fullscreen to true - Stop all background song and run start scene song

Methods

Name	Description
+ StartScene getStartScene(Stage stage)	If instance not null return instance If instance is null, create new startScene
+ void resetStartScene()	Set instance to null

5.12. Package threads

5.12.1. Class BotLogicThread extends Thread

- This class is used as a thread for controlling BaseMonster behaviour

Fields

Name	Description
- BaseScene scene	The scene in the game
- Player player	The player in the game
- boolean isRunning (= true)	Boolean to check whether this thread is running or not
- long delayTime (= 250)	The delay for monsters
- boolean isLaunchWave (= true)	Boolean to check whether the new wave is launched or not
- int currentWaveNumber (= 0)	The current wave number
- int maxWaveNumber	The max wave number in the scene
- boolean isWaveClear (= false)	Boolean to check whether all monsters in the wave are defeated or not
- boolean hasCreatedText (= false)	Boolean to check whether the scene has created the stage status or not

Constructors

Name	Description
+ BotLogicThread(BaseScene scene)	Constructor for BotLogicThread - Set scene and player as scene.getPlayer()

Methods

Name	Description
+ void stopThread()	isRunning = false
+ void run()	Spawn all monsters in each wave and check whether the wave is cleared or not and if all waves are cleared, create the reward zone otherwise create a new wave. Furthermore, it has to check cooldown for each monster's attack and check range from player also
- void checkWave()	If all monsters are defeated but still have a wave, create a new wave but if all wave are defeated, create a reward zone
- void launchWave()	Create a new wave
- void createRewardZone(FightScene scene)	Show Clear! Stage status and call drawRewardArea()
- drawRewardArea(FightScene scene)	Draw a reward depending on the type of FightScene
+ void addWave()	Increasing currentWaveNumber and isLaunchWave = false when isLaunchWave
- void checkCooldownAttack(BaseMonster eachMonster)	Check cooldown for each type of a monster
- void checkBossCooldownAttack(BaseBoss boss)	Check cooldown for each boss' skill
- void dropFromFloatingFloor(BaseBoss boss)	Make the Boss be able to drop down from the floating floor
- void randomUseSkill(BaseBoss	Make the boss uses skill randomly

boss)	
- void chooseSkill(BaseBoss boss)	Boss' Skill Choosing
- void checkGiGeeCooldownAttack(BaseGiGee eachMonster)	Check cooldown for each GiGee' attack
- void createActionIdle(BaseMonster eachMonster)	Create the action while player is not in eachMonster's vision
- void standIdle(BaseMonster eachMonster)	Create the stand animation when player is not in eachMonster's vision
- void walkIdle(BaseMonster eachMonster)	Create the walk animation when player is not in eachMonster's vision
+ void setWaveClear(boolean isWaveClear)	Setter for isWaveClear
+ void setHasCreatedText(boolean hasCreatedText)	Setter for hasCreatedText
+ int getCurrentWaveNumber()	Getter for currentWaveNumber
+ int getMaxWaveNumber()	Getter for maxWaveNumber
+ void setCurrentWaveNumber(int currentWaveNumber)	Setter for currentWaveNumber
+ void setMaxWaveNumber(int waveNumber)	Setter for maxWaveNumber

5.12.2. Class MonsterAnimationThread extends PlayerAnimationThread

- This class is used as a thread for controlling BaseMonster animation

Fields

Name	Description
- ArrayList<States> bossAnimationState (= new ArrayList<>(Arrays.asList(BOSS_SKILL_ONE, BOSS_SKILL_TWO, BOSS_SKILL_THREE_STAND_STILL)))	ArrayList that contains States of the Boss
- int frameCount (= 1)	The amount of frame

- Int speedMultiplier (= 0)	The speed action for boss
- boolean isBossJump (= false)	Boolean to check whether the boss jumps or not

Constructors

Name	Description
+ MonsterAnimationThread(BaseScene scene)	Constructor for MonsterAnimationThread - Call super(scene)

Methods

Name	Description
+ void run()	Creating each monster's animation
- void makeBossFrameAnimation() throws InterruptedException	Creating the boss frames
- void updateBossAnimationFrame(States previousState, States stateNow, BaseBoss boss, Animation bossAnimation)	Updating the boss frames
- void createBossCooldown(BaseBoss boss, States state)	Creating the cooldown timer for each skill
- void updateBossSkillOne(States previousState, States stateNow, BaseBoss boss, Animation bossAnimation)	Updating the animation of the boss' first skill
- void createStompEffect(BaseBoss boss, Animation bossAnimation)	Creating StompEffect for boss
- void updateBossSkillTwoFirstPart(States previousState, States stateNow, BaseBoss boss, Animation bossAnimation)	Updating the animation of the first part of the boss' second skill
- void jumpBossToPlayer(BaseBoss boss, Animation bossAnimation)	Making boss jump to player

- void updateBossSkillTwoLastPart(States previousState, States stateNow, BaseBoss boss, Animation bossAnimation)	Updating the animation of the last part the boss' second skill
- void createStompEffectAndShockWave(BaseBoss boss, Animation bossAnimation)	Creating both StompEffect and ShockWave for boss
- void createMultipleShockWave(BaseBoss boss, Animation bossAnimation)	Call createShockWave() several times
- void createShockWave(BaseBoss boss, Animation bossAnimation, FaceDirection faceDirection, double posX, double posY, int count)	Create two ShockWave objects and add them into the scene and objectsInStage's player
- void updateBossSkillThree(States previousState, States stateNow, BaseBoss boss, Animation bossAnimation)	Updating the animation of the boss' third skill
- Void createFire(BaseBoss boss, Animation bossAnimation)	Create Fire object and add it into the scene and objectsInStage's player

5.12.3. Class MovingThread extends Thread

- This class is used as a thread for controlling entities' movement and objects' movement

Fields

Name	Description
- BaseScene scene	The scene in the game
- boolean isRunning (= true)	Boolean to check whether this thread is running or not
- Player player	The player in the game

Constructors

Name	Description

+ MovingThread(BaseScene scene)	Constructor for MovingThread - Set scene and player as scene.getPlayer()

Methods

Name	Description
+ void stopThread()	isRunning = false
+ void run()	.move() for player, all monsters and objects then updateUI()
- void moveAllMonsters()	.move() each monster inside player.getMonstersInStage()
- void moveAllObjects()	.move() each object inside player.getObjectsInStage()
- void removeObjects(ArrayList<BaseObject> objectsToRemove, ArrayList<Animation> objectsAnimation objectsAnimationToRemove)	Remove all objects and their animations from the scene and player

5.12.4. Class PlayerAnimationThread extends Thread

- This class is used as a thread for controlling player's animation

Fields

Name	Description
# BaseScene scene	The scene in the game
# Player player	The player in the game
# boolean isRunning (= true)	Boolean to check whether this thread is running or not
# ArrayList<States> loopAnimationState (= new ArrayList<>(Arrays.asList(PLAER_STAND_STILL, PLAYER_WALK, MONSTER_STAND_STILL, MONSTER_WALK, PLAYER_JUMP_UP, PLAYER_JUMP_DOWN, PLAYER_DASH, BOSS_JUMP_UP, BOSS_JUMP_DOWN,	The arrayList of States that has to be looped

BOSS_SKILL_THREE_WALK)))	
# ArrayList<States> oneTimeAnimationState (= new ArrayList<>(Arrays.asList(PLAYER_NORMAL_ATTACK, MONSTER_NORMAL_ATTACK, MONSTER_DEAD, MONSTER_SPAWN, BOSS_DEAD)))	The arrayList of States that has to be run only one time
+ int chargeCount (= 0)	The amount of charge for bow user
+ boolean isChargeAttack(= false)	Boolean to check whether player uses charge attack or not
+ boolean isSwitchState (= false)	Boolean to check whether player's states is switched or not

Constructors

Name	Description
+ PlayerAnimationThread(BaseScene scene)	Constructor for PlayerAnimationThread - Set scene and player as scene.getPlayer()

Methods

Name	Description
+ void stopThread()	isRunning = false
+ void run()	Creating player's animation
- void makePlayerFrameAnimation(States stateNow) throws InterruptedException	Creating player's frames
# void oneTimeAnimationUpdateFrame(States previousState, States stateNow, BaseEntity entity, Animation entityAnimation)	Updating the animations that have to be run only one time
# void loopAnimationUpdateFrame(States previousState, States stateNow, Animation entityAnimation)	Updating the animations that have to be looped

- void updatePlayerNormalAttack(States previousState, States stateNow, Animation entityAnimation)	Updating the normal attack animation
- void updatePlayerSpecialAttack(States previousState, States stateNow, BaseEntity entity, Animation entity animation)	Updating the special attack animation
- void updateMonsterDead(States previousState, States stateNow, BaseEntity entity, Animation entity animation)	Updating animations for dead monsters
- void removeMonster(BaseEntity baseEntity, States state)	Waiting for some time then remove the dead entity from the scene and player
- void dropHpMonster()	Creating a HPDrop object and add it into the scene and player
- void updateMonsterAttack(BaseEntity entity)	Updating entity's attacking animation
- void updateArcherGiGeeAttack(BaseEntity entity)	Creating a Missile object and add it into the scene and player
- void updateThrowerGiGeeAttack(BaseEntity entity)	Initialize values for createThrowerBomb()
- void createThrowerBomb(BaseEntity entity, String type, double speed, double yVelocity)	Creating a ThrowerBomb object and add it into the scene and player
# void prepareToUpdateAnimation(States stateNow, Animation entityAnimation)	Set currentFrameIndex to 0 then call updateFrame() and ends with setPreviousState to stateNow
- void updateOneCombo(States previousState, States stateNow, Animation entityAnimation)	Updating the class' animation only the class that has one attack combo
- void createNormalArrow()	Creating a NormalArrowAttack object and add it into the scene and player

- void updateTwoCombo(States previousState, States stateNow, Animation entityAnimation)	Updating the class' animation only the class that has two attack combo
- void playAttackSoundCombo()	Playing the attacking sound (only classes that have two attack combo)
- void checkFrameAttack(Animation entityAnimation)	Call attack() if currentFrameIndex is the attacking index
- void updateBowSpecialAttack(States previousState, States stateNow, Animation entityAnimation)	Creating a Charge Arrow object then if isHoldRMB, trigger updateHoldRMB() otherwise trigger updateReleaseRMB
- void updateHoldRMB(States previousState, States stateNow, Animation entityAnimation)	Updating animations when player holds RMB
- void updateReleaseRMB(States stateNow, Animation entityAnimation, ChargeArrow chargeArrow, int damageIncreasing)	Updating animations when player releases RMB
- void successToChargeAttack(States stateNow, Animation entityAnimation, ChargeArrow chargeArrow, int damageIncreasing)	Adding chargeArrow into the scene and player
- void failToChargeAttack(States stateNow, Animation entityAnimation)	Reset entityAnimation
- void createSpecialAttackCooldownTimer()	Creating an AnimationTimer and makes it as a cooldown timer for special attack
- void updateBroadSwordSpecialAttack(States previousState, States stateNow, Animation entityAnimation)	Updating Animations for broadsword's special attack
- void jumpBroadSword(Animation entityAnimation)	Set values for making broadsword jump
- void	Creating a StompEffect and add it into

createBroadSwordLastAttack(States stateNow, Animation entityAnimation)	the scene and player
- void updateDaggerSpecialAttack(States stateNow, Animation entityAnimation)	Updating Animations for dagger's special attack
- Void createCritRateBuffTimer()	Creating an AnimationTimer and making it as a duration of crit rate increasing

5.12.5. Class StatThread extends Thread

- This class is used as a thread for controlling player stats

Fields

Name	Description
- BaseScene scene	The scene in the game
- Player player	The player in the game
- boolean isRunning (= true)	Boolean to check whether this thread is running or not
- boolean isDashRestored (= false)	Boolean to check whether the dash is restored

Constructors

Name	Description
+ StatThread(BaseScene scene)	Constructor for StatThread - Set scene and player as scene.getPlayer()

Methods

Name	Description
+ void stopThread()	isRunning = false
+ void run()	Activating or Deactivating each perk player obtained then call restoreDash()
- void restoreDash()	Restoring the dash amount
- void getDash()	If the current amount of dash + 1 is still

	not more than the max dash, increase player's dashAmount and update icon's animation

5.13. Package utils

5.13.1. Class AnimationStatus

- This class is used as a container of animation variables

Fields

Name	Description
+ double SPAWN_X (= 1300)	Position on x-axis to spawn
+ double SPAWN_Y (= 500)	Position on y-axis to spawn
+ int START_X (= 4)	Default position on x-axis to set viewport for animation
+ int START_Y (= 25)	Default position on y-axis to set viewport for animation
+ int DEFAULT_ANIMATION_WIDTH (= 184)	Default animation width
+ int DEFAULT_ANIMATION_HEIGHT (= 184)	Default animation height
+ int GAP_WIDTH (= 8)	Default gap width for each animation frame
+ int GAP_HEIGHT (= 8)	Default gap height for each animation frame
+ int DEFAULT_HITBOX_WIDTH (= 46)	Default hitbox width
+ int DEFAULT_HITBOX_HEIGHT (= 92)	Default hitbox height
+ int DEFAULT_HITBOX_X_OFFSET (= 71)	Default hitbox offset on x-axis
+ int DEFAULT_HITBOX_Y_OFFSET (= 63)	Default hitbox offset on y-axis

+ float SCALE (= 0.75f)	Use for adjust size of everything in game except UI to match with different screen size (0.75f for full HD, 1 for 2K)
+ int PIXEL_PER_TILE (= (int)(96 * SCALE))	Size of tile
+ int MOVE_FRAME (= 60)	Moving frame rate
+ int ANIMATION_FRAME (= 8)	Rendering animation frame rate
+ float MOVE_PER_FRAME (= (float) PIXEL_PER_TILE / MOVE_FRAME)	Moving distance per frame based on size of tile and moving frame rate
+ float JUMP_HEIGHT (= 2.5f)	Number of tiles that player can jump
+ float GRAVITY = 1 * SCALE	Gravity for entity
+ float GRAVITY_FOR_OBJECT (= GRAVITY / 3)	Gravity for object
+ double JUMP_VELOCITY (= -14 * Math.sqrt(JUMP_HEIGHT) * SCALE)	Velocity in y-axis when jump
+ int JUMP_FRAME (= 1)	Number of frames in jump state
+ ArrayList<String> DASH_AND_JUMP_WHEN_DEATH_MONSTER (= new ArrayList<>(“PeasantGiGee”, “ArcherGiGee”, “ThrowerGiGee”, “IceThrowerGiGee”))	Arraylist that contain monster’s type that dead animation need to bounce off
+ int ANONYMOUS_STAND_STILL_FRAMES (= 6)	Number of frames in state PLAYER_STAND_STILL for anonymous
+ int ANONYMOUS_WALK_FRAMES (= 5)	Number of frames in state PLAYER_WALK for anonymous
+ int BOW_STAND_STILL_FRAMES (= 6)	Number of frames in state PLAYER_STAND_STILL for bow
+ int BOW_WALK_FRAMES (= 7)	Number of frames in state PLAYER_WALK for bow

+ int BOW_NORMAL_ATTACK_FRAMES (= 7)	Number of frames in state PLAYER_NORMAL_ATTACK for bow
+ int BOW_SPECIAL_ATTACK_FRAMES (= 7)	Number of frames in state PLAYER_SPECIAL_ATTACK for bow
+ int BOW_DASH_FRAMES (= 1)	Number of frames in state PLAYER_DASH for bow
+ int BOW_DEAD_FRAMES (= 0)	Number of frames in state PLAYER_DEAD for bow
+ int BOW_SPECIAL_ATTACK_CHARGE_TIME (= 4)	Minimum frame that can shoot charge arrow in bow's special attack
+ float BOW_SPECIAL_ATTACK_COOLDOWN (= 6)	Cooldown for bow's special attack
+ int BOW_SPECIAL_ATTACK_DAMAGE (= 30)	Extra damage for bow's special attack
+ int BOW_SPECIAL_ATTACK_DAMAGE_INCREASE_PER_FRAME (= 8)	Extra damage for each frame that holds bow's special attack
+ int BOW_SPECIAL_ATTACK_MAX_DAMAGE_INCREASE (= 120)	Maximum extra damage for bow's special attack
+ int BROADSWORD_STAND_STILL_FRAMES (= 6)	Number of frames in state PLAYER_STAND_STILL for broadsword
+ int BROADSWORD_WALK_FRAMES (= 8)	Number of frames in state PLAYER_WALK for broadsword
+ int BROADSWORD_NORMAL_ATTACK_FRAMES (= 15)	Number of frames in state PLAYER_NORMAL_ATTACK for broadsword
+ int BROADSWORD_SPECIAL_ATTACK_FRAMES (= 6)	Number of frames in state PLAYER_SPECIAL_ATTACK for broadsword

+ int BROADSWORD_DASH_FRAMES (= 5)	Number of frames in state PLAYER_DASH for broadsword
+ int BROADSWORD_DEAD_FRAME (= 0)	Number of frames in state PLAYER_DEAD for broadsword
+ int BROADSWORD_ANIMATION_WIDTH (= 472)	Animation width for broadsword
+ int BROADSWORD_ANIMATION_HIGHT (= 400)	Animation height for broadsword
+ int BROADSWORD_HITBOX_WIDTH (= 54)	Hitbox width for broadsword
+ int BROADSWORD_HITBOX_HEIGHT (= 99)	Hitbox height for broadsword
+ int BROADSWORD_HITBOX_X_OFFSET (= 210)	Hitbox offset on x-axis for broadsword
+ int BROADSWORD_HITBOX_Y_OFFSET (= 173)	Hitbox offset on y-axis for broadsword
+ int BROADSWORD_SPECIAL_ATTACK_JUMP_HEIGHT (= -15)	Jump height of broadsword's special attack
+ int BROADSWORD_SPECIAL_ATTACK_COOLDOWN (= 8)	Cooldown for broadsword's special attack
+ int BROADSWORD_SPECIAL_ATTACK_DAMAGE (= 60)	Extra damage for broadsword's special attack
+ int DAGGER_STAND_STILL_FRAMES (= 6)	Number of frames in state PLAYER_STAND_STILL for dagger
+ int DAGGER_WALK_FRAMES (= 3)	Number of frames in state PLAYER_WALK for dagger

+ int DAGGER_NORMAL_ATTACK_FRAMES (= 4)	Number of frames in state PLAYER_NORMAL_ATTACK for dagger
+ int DAGGER_SPECIAL_ATTACK_FRAMES (= 5)	Number of frames in state PLAYER_SPECIAL_ATTACK for dagger
+ int DAGGER_DASH_FRAMES (= 6)	Number of frames in state PLAYER_DASH for dagger
+ int DAGGER_DEAD_FRAMES (= 0)	Number of frames in state PLAYER_DEAD for dagger
+ float DAGGER_SPECIAL_ATTACK_WARP_DISTANCE (= 8)	Warp distance for dagger's special attack
+ float DAGGER_SPECIAL_ATTACK_COOLDOWN (= 6)	Cooldown for dagger's special attack
+ int DAGGER_SPECIAL_ATTACK_DAMAGE (= 60)	Extra damage for dagger's special attack
+ int DAGGER_SPECIAL_ATTACK_CRIT_CHANCE (= 40)	Extra critical rate after using dagger's special attack
+ float DAGGER_SPECIAL_ATTACK_CRIT_BUFF_DURATION (= 2)	Time duration of critical rate buff after using dagger's special attack
+ int SPAWN_FRAMES (= 15)	Number of frames in state MONSER_SPAWN for every monsters
+ int PEASANTGIGEE_STAND_STILL_FRAMES (= 6)	Number of frames in state MONSTER_STAND_STILL for peasantGiGee
+ int PEASANTGIGEE_WALK_FRAMES (= 5)	Number of frames in state MONSTER_WALK for peasantGiGee
+ int PEASANTGIGEE_NORMAL_ATTACK_FRAMES (= 11)	Number of frames in state MONSTER_NORMAL_ATTACK for peasantGiGee
+ int PEASANTGIGEE_DEAD_FRAMES (= 10)	Number of frames in state MONSTER_DEAD for peasantGiGee

+ int KLEEGIGEE_STAND_STILL_FRAMES (= 8)	Number of frames in state MONSTER_STAND_STILL for kleeGiGee
+ int KLEEGIGEE_WALK_FRAMES (= 8)	Number of frames in state MONSTER_WALK for kleeGiGee
+ int KLEEGIGEE_NORMAL_ATTACK_FRAMES (= 0)	Number of frames in state MONSTER_NORMAL_ATTACK for kleeGiGee
+ int KLEEGIGEE_DEAD_FRAMES (= 6)	Number of frames in state MONSTER_DEAD for kleeGiGee
+ int KLEEGIGEE_SPEED_INCREASE (= 20)	Amount of speed increasing when activate bomb
+ float KLEEGIGEE_BOMB_TIME (= 1.25f)	Time duration from activation to explosion of kleeGiGee
+ int ARCHERGIGEE_STAND_STILL_FRAMES (= 6)	Number of frames in state MONSTER_STAND_STILL for archerGiGee
+ int ARCHERGIGEE_WALK_FRAMES (= 5)	Number of frames in state MONSTER_WALK for archerGiGee
+ int ARCHERGIGEE_NORMAL_ATTACK_FRAMES (= 22)	Number of frames in state MONSTER_NORMAL_ATTACK for archerGiGee
+ int ARCHERGIGEE_DEAD_FRAMES (= 10)	Number of frames in state MONSTER_DEAD for archerGiGee
+ int THROWERGIGEE_STAND_STILL_FRAMES (= 4)	Number of frames in state MONSTER_STAND_STILL for thrower GiGee
+ int THROWERGIGEE_WALK_FRAMES (= 4)	Number of frames in state MONSTER_WALK for thrower GiGee
+ int THROWERGIGEE_NORMAL_ATTACK_FRAMES (= 17)	Number of frames in state MONSTER_NORMAL_ATTACK for thrower GiGee

+ int THROWERGIGEE_DEAD_FRAMES (= 8)	Number of frames in state MONSTER_DEAD for throwerGiGee
+ int THROWERGIGEE_HITBOX_WIDTH (= 100)	Hitbox width for throwerGiGee
+ int THROWERGIGEE_HITBOX_X_OFFSET (= 45)	Hitbox offset on x-axis for throwerGiGee
+ float THROWERGIGEE_THROW_TIME (= 2)	Time duration for throwerBomb to reach player position
+ int ICETHROWERGIGEE_STAND_STILL_FRAMES (= 4)	Number of frames in state MONSTER_STAND_STILL for iceThrowerGiGee
+ int ICETHROWERGIGEE_WALK_FRAMES (= 4)	Number of frames in state MONSTER_WALK for iceThrowerGiGee
+ int ICETHROWERGIGEE_NORMAL_ATTACK_FRAMES (= 17)	Number of frames in state MONSTER_NORMAL_ATTACK for iceThrowerGiGee
+ int ICETHROWERGIGEE_DEAD_FRAMES (= 8)	Number of frames in state MONSTER_DEAD for iceThrowerGiGee
+ float ICETHROWERGIGEE_THROW_TIME (= 1.5f)	Time duration for throwerBomb to reach player position
+ int ICETHROWERGIGEE_SLOW_AMOUNT (= 40)	Amount of slow effect from iceThrowerGiGee’s throwerBomb
+ int ICETHROWERGIGEE_MAX_SLOW_AMOUNT (= 80)	Maximum amount of slow effect that can stack in player from iceThrowerGiGee’s throwerBomb
+ int ICETHROWERGIGEE_SLOW_DURATION (= 2)	Time duration of slow effect from iceThrowerGiGee’s throwerBomb
+ int BIGBOY_STAND_STILL_FRAMES (= 1)	Number of frames in state MONSTER_STAND_STILL for bigBoy

+ int BIGBOY_WALK_FRAMES (= 10)	Number of frames in state MONSTER_WALK for bigBoy
+ int BIGBOY_SKILL_ONE_FRAMES (= 10)	Number of frames in state BOSS_SKILL_ONE for bigBoy
+ ArrayList<Integer> BIGBOY_SKILL_ONE_FRAMES_HOLD (= new ArrayList<>(Arrays.asList(1,3,3,5,1,1,1,2,1,1)))	ArrayList that contain number of frames' duration to hold each frame in bigBoy's skill one
+ float BIGBOY_SKILL_ONE_ANIMATION_SPEED (= 2)	Speed multiplier of bigBoy's skill one
+ int BIGBOY_SKILL_TWO_FRAMES (= 7)	Number of frames in state BOSS_SKILL_TWO for bigBoy
+ ArrayList<Integer> BIGBOY_SKILL_TWO_FRAMES_HOLD (= new ArrayList<>(Arrays.asList(1,3,3,1,1,1,1)))	ArrayList that contain number of frames' duration to hold each frame in bigBoy's skill two
+ float BIGBOY_SKILL_TWO_ANIMATION_SPEED (= 1.5f)	Speed multiplier of bigBoy's skill two
+ int BIGBOY_SKILL_THREE_STAND_FRAMES (= 5)	Number of frames in state BOSS_SKILL_THREE_STAND_STILL for bigBoy
+ int BIGBOY_SKILL_THREE_WALK_FRAMES (= 10)	Number of frames in state BOSS_SKILL_THREE_WALK for bigBoy
+ int BIGBOY_DEAD_FRAMES (= 1)	Number of frames in state BOSS_DEAD for bigBoy
+ int BIGBOY_ANIMATION_WIDTH (= 394)	Animation width
+ int BIGBOY_ANIMATION_HEIGHT (= 304)	Animation height

+ int BIGBOY_HITBOX_WIDTH (= 212)	Hitbox width
+ int BIGBOY_HITBOX_HEIGHT (= 207)	Hitbox height
+ int BIGBOY_HITBOX_X_OFFSET (= 68)	Hitbox offset on x-axis for bigBoy
+ int BIGBOY_HITBOX_Y_OFFSET (= 91)	Hitbox offset on y-axis for bigBoy

5.13.2. Class CanvasDrawer extends Canvas

- This class is used as a drawer

Fields

Name	Description
- Image template (= new Image(ClassLoader.getSystemResource("background/StageTemplate.png").toString()))	Template for drawing platform
- BaseScene scene	The scene to draw

Constructors

Name	Description
+ CanvasDrawer(BaseScene scene, double width, double height)	Constructor for CanvasDrawer - call super(width, height) and set scene to scene

Methods

Name	Description
- int getValueInThatCoor(int x, int y)	return the number inside scene's levelData at the specific index
- int getXCoor(int x, int y)	use getValueInThatCoor() to bring a number then checking it with template to find a row number
- int getYCoor(int x, int y)	use getValueInThatCoor() to bring a number then checking it with template to find a column number

+ void drawBackground(GraphicsContext gc)	draw background in the game by combining a part of template at the getXCoor() number and the getYCoor() number
- void drawDoors(ArrayList<String> randomRoomTypes, int x, int y)	create door objects in the game and add it into the scene
+ void drawMinimap(GraphicsContext gc)	draw background of the minimap in the game (having an option to adjust the colors of the image based on the background color)
+ void drawEntities(GraphicsContext gc)	clearRect() gc then call drawDoorsMinimap(), drawPlayerMinimap() and drawAllMonstersMinimap()
- void drawDoorsMinimap(GraphicsContext gc)	draw door objects in the minimap
- void drawPlayerMinimap(GraphicsContext gc)	draw player in the minimap
- void drawAllMonstersMinimap(GraphicsContext gc)	draw all monsters in the minimap

5.13.3. Class GameConstants

- This class is used as a container of the game variables

Fields

Name	Description
+ int gameWidth	Screen width
+ int gameHeight	Screen height
+ int uiOffset	Offset for UI
+ int healthWidth	Player's health bar width
+ int healthHeight	Player's health bar height
+ int healthUIWidth	HealthUI width

+ int healthUIHeight	HealthUI height
+ double scaleForUI	Use for adjust size of UI to match with different screen size
+ double edgeRightOfHealthBar	Position on x-axis of edge right of health bar
+ double edgeTopOfHealthBar	Position on y-axis of edge top of health bar
+ ArrayList<String> ROOM_TYPES (= new ArrayList<>(Arrays.asList("HP", "Coin", "PerkChest")));	ArrayList that contain room's type
+ double borderL (= 0)	Position in x-axis of left border
+ double borderR (= borderL + gameWidth)	Position in x-axis of right border
+ SoundLoader START_SCENE_SONG (= new SoundLoader("StartSceneSong", "sfx/sample_startscene_song.mp3"))	Song for StartScene
+ SoundLoader CLASS_SELECTION_SCENE_SONG = (new SoundLoader("ClassSelectionSceneSong","sfx/sample_classselectionscene_song.mp3"))	Song for ClassSelectionScene
+ SoundLoader FIGHT_SCENE_SONG1 (= new SoundLoader("FightSceneSong1", "sfx/sample_fightscene_song1.mp3"))	Song #1 for FightScene
+ SoundLoader FIGHT_SCENE_SONG2 (= new SoundLoader("FightSceneSong2", "sfx/sample_fightscene_song2.mp3"))	Song #2 for FightScene
+ SoundLoader FIGHT_SCENE_SONG3 (= new SoundLoader("FightSceneSong3	Song #3 for FightScene

" "sfx/sample_fightscene_song3.mp3"))	
+ SoundLoader FIGHT_SCENE_SONG4 (= new SoundLoader("FightSceneSong4" "sfx/sample_fightscene_song4.mp3"))	Song #4 for FightScene
+ SoundLoader FIGHT_SCENE_SONG5 (= new SoundLoader("FightSceneSong5" "sfx/sample_fightscene_song5.mp3"))	Song #5 for FightScene
+ SoundLoader FIGHT_SCENE_SONG6 (= new SoundLoader("FightSceneSong6" "sfx/sample_fightscene_song6.mp3"))	Song #6 for FightScene
+ SoundLoader MARKET_SCENE_SONG (= new SoundLoader("MarketSceneSong", "sfx/sample_marketscene_song.mp3"))	Song for MarketScene
+ SoundLoader BOSS_SCENE_SONG (= new SoundLoader("BossSceneSong", "sfx/sample_bossscene_song.mp3"))	Song for BossScene
+ SoundLoader INVENTORY (= new SoundLoader("Inventory", "sfx/sample_inventory.mp3"));	Sound used when player opens his inventory
+ ArrayList<Integer> passableTile (= new ArrayList<>(Arrays.asList(48,49,50,51,52)))	ArrayList containing numbers that refer to a floor that can be passed through
+ ArrayList<Integer> airTile (= new ArrayList<>(Arrays.asList(-9, -4, -3, -2, -1, 11, 97, 98, 99)))	ArrayList containing numbers for creating something and after creating, it will be the air in the game

+ ArrayList<Integer> spawnMonsterTile = new ArrayList<>(Arrays.asList(-9, -4, -3, -2, 98));	ArrayList containing numbers for spawning monsters
+ Font thaleahFatFont (= Font.loadFont(ClassLoader.getSystemResourceAsStream("fonts/ThaleahFat.ttf"), 14))	Font used in inventory
+ Font thaleahFatFontButton (= Font.loadFont(ClassLoader.getSystemResourceAsStream("fonts/ThaleahFat.ttf"), 90))	Font used when we need to create a button
+ Color GREEN_FOR_UI (= Color.rgb(0, 144,104))	Color used in inventory
+ double soundVolume (= 0.05)	Constant for adjusting the sound volume

Methods

Name	Description
+ void setGameWidth(int gameWidth)	set gameWidth, healthWidth, healthUIWidth and edgeRightOfHealthBar
+ void setGameHeight(int gameHeight)	set gameHeight, healthHeight, healthUIHeight, edgetopOfHealthBar, uiOffset and scaleForUI
+ void stopAllBackgroundSongs()	Stop all background songs in all scenes

5.13.4. Class HelperMethods

- This class is used as a container of some methods that are frequently used in multiple classes

Fields

Name	Description
- Random random = new Random()	A randomizer
- SoundLoader PLAYER_WALK (= new SoundLoader("PlayerWalk", "sfx/sample_walk.mp3"))	Sound used when player walks

- SoundLoader PLAYER_DASH (= new SoundLoader("PlayerDash", "sfx/sample_dash.wav"))	Sound used when player uses dash
- SoundLoader PLAYER_JUMP1 (= new SoundLoader("PlayerJump1", "sfx/sample_jump1.mp3"))	Sound used when player jumps
- SoundLoader PLAYER_JUMP2 (= new SoundLoader("PlayerJump2", "sfx/sample_jump2.mp3"))	Sound used when player jumps again
- SoundLoader PLAYER_COLLECT (= new SoundLoader("PlayerCollect", "sfx/sample_collect.mp3"))	Sound used when player collects a perk
- SoundLoader PLAYER_UPGRADE = new SoundLoader("PlayerUpgrade", "sfx/sample_upgrade.mp3")	Sound used when player upgrade a perk
- SoundLoader PLAYER_BEING_HIT (= new SoundLoader("PlayerHurt", "sfx/sample_player_hurt.mp3"))	Sound used when player received damage but not die
- SoundLoader PLAYER_DODGE (= new SoundLoader("PlayerDodge", "sfx/sample_miss.mp3"))	Sound used when player can evade the attack
- SoundLoader PLAYER_BUY_SUCCESS = new SoundLoader("BuyingSuccess", "sfx/sample_buying_success.mp3")	Sound used when player buys successfully
- SoundLoader PLAYER_BUY_FAIL (= new SoundLoader("BuyingFail", "sfx/sample_buying_fail.mp3"))	Sound used when player fails to buy
- SoundLoader PLAYER_BUY_MISS (= new SoundLoader("BuyingMiss", "sfx/sample_buying_miss.wav"))	Sound used when player buys and TIER2_2 DiscountMasterPerk's chance is activated
- SoundLoader PLAYER_BRUH (=	Sound used when player clicks at the

new SoundLoader("PlayerBruh", "sfx/sample_bruh.mp3")	TIER2_1 or TIER2_2 perk in the BlackSmith
- SoundLoader PLAYER_DEAD = new SoundLoader("PlayerDead", "sfx/sample_player_dead.mp3")	Sound used when player dies
- SoundLoader MONSTER_BEING_HIT (= new SoundLoader("MonsterHurt", "sfx/sample_monster_hurt.mp3"))	Sound used when a monster receives a damage and survives
- SoundLoader MONSTER_DEAD (= new SoundLoader("MonsterDead", "sfx/sample_monster_dead.mp3"))	Sound used when a monster dies
- SoundLoader MONSTER_DODGE (= new SoundLoader("MonsterDodge", "sfx/sample_miss.mp3"))	Sound used when a monster dodges an attack
- SoundLoader BOSS_FIRE (= new SoundLoader("BossFire", "sfx/sample_flame.mp3"))	Sound used when a boss launches a fire attack
- SoundLoader BOSS_STOMP_EFFECT (= new SoundLoader("BossStomping", "sfx/sample_stomp.mp3"))	Sound used when a boss launches a stomping attack
- SoundLoader DAGGER_COMBO_ATTACK1 (= new SoundLoader("DaggerComboAttack1", "sfx/sample_dagger_combo1.mp3"))	Sound used when a dagger class uses normal attack
- SoundLoader DAGGER_COMBO_ATTACK2 (= new SoundLoader("DaggerComboAttack2", "sfx/sample_dagger_combo2.mp3"))	Sound used when a dagger class uses normal attack again
- SoundLoader BROADSWORD_ATTACK (= new	Sound used when a broadsword class uses normal attack and the first 2 animation of special attack

SoundLoader("BroadSwordAttack", "sfx/sample_broadsword_attack.mp3"))	
- SoundLoader BROADSWORD_STOMP_EFFECT (= new SoundLoader("BroadSwordStomping", "sfx/sample_stomp.mp3"))	Sound used when a broadsword class is at the last animation of the special attack
- SoundLoader NORMAL_ARROW_ATTACK (= new SoundLoader("NormalArrowAttack", "sfx/sample_normal_bow_attack.mp3"))	Sound used when a bow clas uses a normal attack
- SoundLoader CHARGE_ARROW_CHARGE (= new SoundLoader("ChargeArrowCharging", "sfx/sample_charge_bow_charge.mp3"))	Sound used when a bow class charges the arrow in his special attack
- SoundLoader CHARGE_ARROW_ATTACK (= new SoundLoader("ChargeArrowAttack", "sfx/sample_charge_bow_attack.mp3"))	Sound used when a bow class launches the charged arrow
- SoundLoader THROWERBOMB_ATTACK (= new SoundLoader("ThrowerBombAttack", "sfx/sample_throwerbomb.wav"))	Sound used when a ThrowerBomb explodes
- SoundLoader KLEE_BOMB = new SoundLoader("KleeBomb", "sfx/sample_kleebomb.mp3")	Sound used when KleeGiGee explodes himself
- SoundLoader NUCLEAR_BOMB (= new SoundLoader("NuclearBomb", "sfx/sample_nuclearbomb.mp3"))	Sound used when the boss explodes himself

- SoundLoader GET_HEAL = new SoundLoader("GetHeal", "sfx/sample_getheal.mp3")	Sound used when player obtains HPDrop
- SoundLoader MERCHANT_SELL (= new SoundLoader("MerchantSell", "sfx/sample_merchant.mp3"))	Sound used when player purchases in Merchant
- SoundLoader PERKCHEST_GET = new SoundLoader("PerkChestOpen", "sfx/sample_perkchest.mp3")	Sound used when player activates PerkChestReward
- SoundLoader BASEREWARD_GET = new SoundLoader("BaseRewardGet", "sfx/sample_basereward.mp3")	Sound used when player obtains BaseReward
- SoundLoader DOOR_OPEN = new SoundLoader("DoorOpen", "sfx/sample_dooropen.mp3")	Sound when player interacts with the open doors

Methods

Name	Description
+ int randomInteger(int start, int end)	Return an integer from start to end
+ float randomFloat(float start, float end)	Return a float between start and end
+ float randomChance()	Return a float between 0 and 1
+ int randomIntegerUsingPercent(int value, int percent)	Return a float between (1 - percent) * value and (1.0 + percent) * value
+ int finalValuePercent(int base, int percent)	Return (1.0 + percent) * base
+ float finalValuePercentFloat(float base, int percent)	Return (1.0 + percent) * base
+ int calculateActionAttack(BaseEntity attacker, BaseEntity attacked, boolean isCrit, int extraDamagePercent)	The damage calculation system.

+ int calculateReflectionAttack(int damageReflect, BaseMonster attacked)	The damage reflection calculation system.
+ void removeFromUnobtain(Player player, BasePerk removePerk)	Remove a perk from player's notObtainPerk
+ ArrayList<BasePerk> generateAllPerks(Player player)	Generates all perks in the game
+ Button createButton(String text)	Return createButton(text, null, BLACK)
+ Button createButton(String text, Color bgColor, Color textColor)	Return the created button
+ Text createTextForInventory(String msg, int fontSize, Color color, double posX, double posY)	Return the text used in inventory
+ Text createText(String msg, int fontSize, Color color)	Return createText(msg, fontSize, color, BOLD)
+ Text createText(String msg, int fontSize, Color fontColor, FontWeight fontWeight)	Return the created text
+ boolean canMoveHere(BaseEntity entity, double x, double y, double width, double height, int[][] lvlData)	Return true if the entity can move to this tile otherwise return false
- boolean isSolid(BaseEntity entity, double x, double y, int[][] lvlData)	Return true if the tile is a solid tile for entity otherwise return false
+ boolean canMoveHereForObject(double x, double y, double width, double height, int[][] lvlData)	Return true if the object can move to this tile otherwise return false
- boolean isSolidForObject(double x, double y, int[][] lvlData)	Return true if the tile is a solid tile for object otherwise return false
+ void spawnPlayer(Player player, int[][] lvlData)	Spawn player at the specific position
+ int getTypeOfTile(double x, double y, int[][] lvlData)	Return the number at the specific index to check the type of the tiles
+ void spawnMonsters(int[][] lvlData, BaseScene baseScene)	Spawn all monsters in the scene's levelData

- void spawnMonster(Player player, String monsterType, BaseScene baseScene, double xPos, double yPos)	Spawn a specific monster at the specific position
+ ArrayList<SoundLoader> loadEntitiesSound(BaseEntity baseEntity)	Return the arrayList containing specific sounds for each entity
+ ArrayList<SoundLoader> loadObjectsSound(BaseObject baseObject)	Return the arrayList containing specific sounds for each object
+ SoundLoader getSoundFromSoundList(ArrayList<SoundLoader> soundList, String name)	Return the specific sound inside the soundList
+ String randomRoomType()	Return a random String in ROOM_TYPES
+ calculateNewHp(BaseEntity baseEntity, int maxHpPercent, int healPercent)	To Calculate new CurrentHp and new MaxHp
+ int countMonstersNearby(Player player, float tilesX, float tilesY)	To count the amount of monsters around player
+ boolean hasPerk(Player player, BasePerk basePerk, boolean useTier)	To Check whether player has this perk or not
+ ArrayList<BasePerk> createPerkChoices(Player player, ArrayList<BasePerk> toRandom, int number)	Randoming perks that aren't obtained and creating the perk UI for player to choose a perk
- void addPerkToResult(BasePerk selectedPerk, ArrayList<BasePerk> result)	Add a perk that is not in the result to result
+ ArrayList<String> createDoorChoices(ArrayList<String> toRandom, int number)	Return the arrayList containing doorType's String to create doors
+ void createColumns(GridPane gridPane, int[] array)	Adding specific column constraints into gridPane
+ void createRows(GridPane gridPane, int[] array)	Adding specific row constraints into gridPane
+ ArrayList<SoundLoader>	Return the arrayList of all sounds in the

getAllSounds()	game
+ void addAndDeleteText(BaseEntity baseEntity, BaseScene scene, Text text)	Create damage texts and reflection damage texts in the game that will be removed
+ void addAndDeleteStageStatus(BaseScene scene, Text text)	Create a stage status in the game that will be remove
+ knockBack(BaseEntity baseEntity, FaceDirection direction, BaseScene scene)	Knockback System
+ void immuneAfterAttacked(BaseEntity baseEntity)	baseEntity will be immuned for a time
+ void resetAllScenes()	Reset all scenes in the game
+ void createSymbol(BaseScene baseScene, String type, double posX, double posY, BasePerk basePerk)	Create symbol objects and add into the game

5.13.5. Class LevelData

- This class is used as a container of level datas in the game

Fields

Name	Description
+ int NUMBER_OF_FIGHT_ROOM_BEFORE_MARKET (= 5)	Number of FightScene before going to MarketScene
+ int[][] classSelectionSceneData	Please Read in this class (these are level data of each scene in game)
+ int[][] firstFightSceneData	
+ int[][] secondFightSceneData	
+ int[][] thirdFightSceneData	
+ int[][] forthFightSceneData	
+ int[][] fifthFightSceneData	
+ int[][] marketSceneData	

+ int[][] BossSceneData	

Methods

Name	Description
+ ArrayList<int[][]> getAllFightScene()	Return the arrayList of all fightSceneDatas

5.13.6. Class MonsterStatus

- This class is used as a container of monster stats

Fields

Name	Description
+ float SPAWN_TIME (= 2.5f)	Time duration for monster to spawn
+ int MIN_WALKTIME (= 2)	Minimum time duration for monster to walk when idle
+ int MAX_WALKTIME (= 5)	Maximum time duration for monster to walk when idle
+ int MONSTER_DROP_VARIATION (= 30)	For create variation of coin drop from monster
+ float TURNING_DELAY (= 1.5f)	Time duration for monster to turn around
+ float KLEE_TURNING_DELAY (= 0.75f)	Time duration for kleeGiGee to turn around
+ float LOW_MOVEMENT_SPEED (= 1.5f)	Low movement speed
+ float MEDIUM_MOVEMENT_SPEED (= 2.5f)	Medium movement speed
+ float HIGH_MOVEMENT_SPEED (= 4f)	High movement speed
+ int LOW_EVADE_RATE (= 5)	Low evade rate
+ int MEDIUM_EVADE_RATE (= 10)	Medium evade rate

+ float SHORT_VISION (= 8f)	Short vision
+ float MEDIUM_VISION (= 12f)	Medium vision
+ float CLOSED_RANGE (= 0.8f)	Closed attack range
+ float MID_RANGE (= 2f)	Mid attack range
+ float LONG_RANGE (= 8f)	Long attack range
+ float GIGEE_DROP_RATE (= 0.15f)	Drop rate from GiGee
+ float BOSS_DROP_RATE (= 0.1f)	Drop rate from Boss
+ int COIN_DROP_GIGEE (= 5)	Amount of coins drop from GiGee
+ int COIN_DROP_BOSS (= 300)	Amount of coins drop from Boss
+ int LOW_KNOCKBACK_CHANCE (= 5)	Low knockback chance
+ int HIGH_KNOCKBACK_CHANCE (= 60)	High knockback chance
+ float LOW_KNOCKBACK_DISTANCE (= 0.1f)	Low knockback distance
+ float MEDIUM_KNOCKBACK_DISTANCE (= 0.15f)	Medium knockback distance
+ float HIGH_KNOCKBACK_DISTANCE (= 0.2f)	High knockback distance
+ int PEASANT_HP (= 80)	PeasantGiGee’s health
+ int PEASANT_ATK (= 24)	PeasantGiGee’s attack damage
+ int PEASANT_DEF (= 6)	PeasantGiGee’s defense
+ int PEASANT_ATK_DELAY (= 3)	PeasantGiGee’s attack delay
+ int KLEE_HP (= 30)	KleeGiGee’s health
+ int KLEE_ATK (= 50)	KleeGiGee’s attack damage
+ int KLEE_DEF (= 0)	KleeGiGee’s defense

+ float KLEE_ATK_DELAY (= 6)	KleeGiGee's attack delay
+ int ARCHER_HP (= 40)	ArcherGiGee's health
+ int ARCHER_ATK (= 20)	ArcherGiGee's attack damage
+ int ARCHER_DEF (= 2)	ArcherGiGee's defense
+ float ARCHER_ATK_DELAY (= 6)	ArcherGiGee's attack delay
+ int THROWER_HP (= 40)	ThrowerGiGee's health
+ int THROWER_ATK (= 20)	ThrowerGiGee's attack damage
+ int THROWER_DEF (= 2)	ThrowerGiGee's defense
+ float THROWER_ATK_DELAY (= 8)	ThrowerGiGee's attack delay
+ int ICE_THROWER_ATK (= 10)	IceThrowerGiGee's attack damage
+ int BIGBOY_HP (= 800)	BigBoy's health
+ int BIGBOY_ATK (= 50)	BigBoy's attack damage
+ int BIGBOY_DEF (= 15)	BigBoy's defense
+ float BIGBOY_SKILL_ONE_COOLDOWN (= 1.5f)	BigBoy's skill one cooldown
+ float BIGBOY_COOLDOWN_AFTER_SKILL_ONE (= 1.25f)	BigBoy's cooldown after using skill one
+ float BIGBOY_SKILL_TWO_COOLDOWN (= 6)	BigBoy's skill two cooldown
+ float BIGBOY_COOLDOWN_AFTER_SKILL_TWO (= 3.5f)	BigBoy's cooldown after using skill two
+ float BIGBOY_SKILL_THREE_COOLDOWN (= 6)	Bigboy's skill three cooldown
+ float BIGBOY_COOLDOWN_AFTER_SKILL_THREE (= 6)	BigBoy's cooldown after using skill three
+ float	BigBoy's attack range for skill one

BIGBOY_SKILL_ONE_ATTACK_RANGE (= 1.5f)	
+ float BIGBOY_SKILL_TWO_ATTACK_RANGE (= 8)	BigBoy's attack range for skill two
+ final float BIGBOY_SKILL_TWO_JUMP_TIME (= 1.5f)	Time duration for jump to reach player position
+ float BIGBOY_SKILL_TWO_MAX_JUMP_DISTANCE (= 15)	Maximum distance that BigBoy can jump on x-axis
+ int BIGBOY_SKILL_TWO_NUMBERS_OF_SHOCKWAVE (= 5)	Number of shockwave release after using skill two
+ float BIGBOY_SKILL_THREE_ATTACK_RANGE (= 8)	BigBoy's attack range for skill three
+ float BIGBOY_SKILL_THREEE_MIN_ATTACK_RANGE (= 1)	Minimum attack range for BigBoy's skill three
+ float BIGBOY_NUCLEAR_TIME (= 1.5f)	The amount of time that the Bigboy's explosion stays

5.13.7. Class ObjectStatus

- This class is used as a container of object data

Fields

Name	Description
+ int NOT_MOVE (= 0)	Object's speed is 0
+ int NORMAL_ARROW_WIDTH (= 81)	Normal arrow's width
+ int NORMAL_ARROW_HEIGHT (= 21)	Normal arrow's height
+ int NORMAL_ARROW_SPEED (= 12)	Normal arrow's speed
+ int CHARGE_ARROW_WIDTH (= 93)	Charge arrow's width

+ int CHARGE_ARROW_HEIGHT (= 27)	Charge arrow's height
+ int CHARGE_ARROW_SPEED (= 24)	Charge arrow's speed
+ int MISSILE_WIDTH (= 129)	Missile's width
+ int MISSILE_HEIGHT (= 27)	Missile's height
+ int MISSILE_SPEED (= 18)	Missile's speed
+ int HP_DROP_LARGE_WIDTH (= 45)	Hp drop large's width
+ int HP_DROP_LARGE_HEIGHT (= 45)	Hp drop large's height
+ float HP_DROP_LARGE_HEAL (= 35)	Hp drop large's heal percent
+ int HP_DROP_SMALL_WIDTH (= 27)	Hp drop small's width
+ int HP_DROP_SMALL_HEIGHT (= 36)	Hp drop small's height
+ float HP_DROP_SMALL_HEAL (= 5)	Hp drop small's heal percent
+ int SYMBOL_WIDTH (= 140)	Symbol's width
+ int SYMBOL_HEIGHT (= 140)	Symbol's height
+ int PERK_SYMBOL_WIDTH (= 96)	Perk Symbol's width
+ int PERK_SYMBOL_HEIGHT (= 99)	Perk Symbol's height
+ int COIN_REWARD_WIDTH (= 288)	Coin reward's width
+ int COIN_REWARD_HEIGHT (= 114)	Coin reward's height
+ Int COIN_REWARD (= 200)	Amount of coins from coin reward
+ int HP_REWARD_WIDTH (= 252)	Hp reward's width
+ int HP_REWARD_HEIGHT (= 144)	Hp reward's height

+ Int HP_REWARD_MAX_HP_INCREASE (= 25)	Amount of max hp increase from hp reward
+ Int HP_REWARD_HEAL (= 50)	Amount of heal from hp reward
+ int PERK_CHEST_REWARD_WIDTH (= 106)	Perk chest reward's width
+ int PERK_CHEST_REWARD_HEIGHT (= 94)	Perk chest reward's height
+ int DOOR_WIDTH (= 291)	Door's width
+ int DOOR_HEIGHT (= 309)	Door's height
+ int KLEE_BOMB_WIDTH (= 562)	Klee Bomb's width
+ int KLEE_BOMB_HEIGHT (= 400)	Klee Bomb's height
+ int KLEE_BOMB_HITBOX_X_OFFSET (= 165)	Klee bomb's hit box offset on x-axis
+ int KLEE_BOMB_HITBOX_Y_OFFSET (= 235)	Klee bomb's hit box offset on y-axis
+ int KLEE_BOMB_HITBOX_WIDTH (= 250)	Klee bomb's hit box width
+ int KLEE_BOMB_HITBOX_HEIGHT (= 105)	Klee bomb's hit box height
+ int BOMB_FRAMES (= 10)	bomb's frames
+ int KLEE_BOMB_FRAMES_PER_SECOND (= 12)	Klee bomb's frames per second
+ int NUCLEAR_BOMB_WIDTH (= 1015)	Nuclear bomb's width
+ int NUCLEAR_BOMB_HEIGHT (= 595)	Nuclear bomb's height
+ int	Nuclear bomb's offset in x-axis

	NUCLEAR_BOMB_HITBOX_X_OFFSET (= 160)	
+	int NUCLEAR_BOMB_HITBOX_Y_OFFSET (= 190)	Nuclear bomb's offset in y-axis
+	int NUCLEAR_BOMB_HITBOX_WIDTH (= 670)	Nuclear bomb's hitbox width
+	int NUCLEAR_BOMB_HITBOX_HEIGHT (= 310)	Nuclear bomb's hitbox height
+	int NUCLEAR_BOMB_FRAMES_PER_SECOND (= 8)	Nuclear bomb's frames per second
+	int THROWER_BOMB_WIDTH (= 45)	Thrower bomb's width
+	int THROWER_BOMB_HEIGHT (= 45)	Thrower bomb's height
+	int THROWER_BOMB_ACTIVATE_WIDTH (= 562)	Thrower bomb's activate width
+	int THROWER_BOMB_ACTIVATE_HEIGHT (= 454)	Thrower bomb's activate height
+	int THROWER_BOMB_ACTIVATE_HITBOX_X_OFFSET (= 165)	Thrower bomb's activate hit box offset on x-axis
+	int THROWER_BOMB_ACTIVATE_HITBOX_Y_OFFSET (= 142)	Thrower bomb's activate hit box offset on y-axis
+	int THROWER_BOMB_ACTIVATE_HITBOX_WIDTH (= 208)	Thrower bomb's activate hit box width
+	int THROWER_BOMB_ACTIVATE_HITBOX_HEIGHT (= 182)	Thrower bomb's activate hit box height
+	int THROWER_BOMB_ACTIVATE_FRAMES (= 10)	Thrower bomb's activate frames

+ int THROWER_BOMB_ACTIVATE_FRAMES_PER_SECOND (= 18)	Thrower bomb's activate frames per second
+ int THROWER_ICE_BOMB_ACTIVATE_WIDTH (= 586)	Thrower ice bomb's activate width
+ int THROWER_ICE_BOMB_ACTIVATE_HEIGHT (= 400)	Thrower ice bomb's activate height
+ int THROWER_ICE_BOMB_ACTIVATE_HITBOX_X_OFFSET (= 145)	Thrower ice bomb's activate hitbox offset on x-axis
+ int THROWER_ICE_BOMB_ACTIVATE_HITBOX_Y_OFFSET (= 170)	Thrower ice bomb's activate hitbox offset on y-axis
+ int THROWER_ICE_BOMB_ACTIVATE_HITBOX_WIDTH (= 280)	Thrower ice bomb's activate hitbox width
+ int THROWER_ICE_BOMB_ACTIVATE_HITBOX_HEIGHT (= 110)	Thrower ice bomb's activate hitbox height
+ int THROWER_ICE_BOMB_ACTIVATE_FRAMES (= 10)	Thrower ice bomb's activate frames
+ int THROWER_ICE_BOMB_ACTIVATE_FRAMES_PER_SECOND (= 18)	Thrower ice bomb's activate frames per second
+ int STOMP_EFFECT_WIDTH (= 628)	Stomp effect's width
+ int STOMP_EFFECT_HEIGHT (= 221)	Stomp effect's height
+ int STOMP_EFFECT_HITBOX_X_OFFSET (= 80)	Stomp effect's hitbox offset on x-axis
+ int STOMP_EFFECT_HITBOX_Y_O	Stomp effect's hitbox offset on y-axis

FFSET (= 120)	
+ int STOMP_EFFECT_HITBOX_WIDTH (= 460)	Stomp effect's hitbox width
+ int STOMP_EFFECT_HITBOX_HEIGHT (= 80)	Stomp effect's hitbox height
+ int STOMP_EFFECT_FRAMES (= 3)	Stomp effect's frames
+ int STOMP_EFFECT_FRAMES_PER_SECOND (= 12)	Stomp effect's frames per second
+ int SHOCK_WAVE_WIDTH (= 313)	Shock wave's width
+ int SHOCK_WAVE_HEIGHT (= 367)	Shock wave's height
+ int SHOCK_WAVE_HITBOX_X_OFFSET (= 83)	Shock wave's hitbox offset on x-axis
+ int SHOCK_WAVE_HITBOX_Y_OFFSET (= 133)	Shock wave's hitbox offset on y-axis
+ int SHOCK_WAVE_HITBOX_WIDTH (= 130)	Shock wave's hitbox width
+ int SHOCK_WAVE_HITBOX_HEIGHT (= 195)	Shock wave's hitbox height
+ int SHOCK_WAVE_FRAMES (= 9)	Shock wave's frames
+ int SHOCK_WAVE_FRAMES_PER_SECOND (= 12)	Shock wave's frames per second
+ int SHOCK_WAVE_EXTRA_DAMAGE (= -60)	Extra damage for shock wave
+ int FIRE_WIDTH (= 1150)	Fire's width

+ int FIRE_HEIGHT (= 610)	Fire's height
+ int FIRE_HITBOX_X_OFFSET (= 120)	Fire's hitbox offset on x-axis
+ int FIRE_HITBOX_Y_OFFSET (= 240)	Fire's hitbox offset on y-axis
+ int FIRE_HITBOX_WIDTH (= 710)	Fire's hitbox width
+ int FIRE_HITBOX_HEIGHT (= 225)	Fire's hitbox height
+ int FIRE_FRAMES (= 14)	Fire's frames
+ int FIRE_FRAMES_PER_SECOND (= 10)	Fire's frames per second
+ int FIRE_EXTRA_DAMAGE (= -85)	Extra damage for fire
+ int FIRE_TIME (= 4)	Fire's time duration
+ int VENDING_MACHINE_WIDTH (= 511)	Vending machine's width
+ int VENDING_MACHINE_HEIGHT (= 400)	Vending machine's height
+ int VENDING_MACHINE_FRAMES (= 20)	Vending machine's frames
+ int VENDING_MACHINE_FRAMES_PER_SECOND (= 12)	Vending machines's frames per second
+ int SHROOM_WIDTH (= 94)	Shroom's width
+ int SHROOM_HEIGHT (= 94)	Shroom's height
+ int SHROOM_FRAMES (= 4)	Shroom's frames
+ int SHROOM_FRAMES_PER_SECOND (= 8)	Shroom's frames per second
+ int MERCHANT_WIDTH (= 172)	Merchant's width

+ int MERCHANT_HEIGHT (= 157)	Merchant's height
+ int MERCHANT_HEAL_PERCENT (= 40)	Amount of heal from merchant
+ int MIN_POTION_PRICE (= 100)	Minimum potion price
+ int MAX_POTION_PRICE (= 120)	Maximum potion price
+ ArrayList<Integer> BLACK_SMITH_WIDTH (= new ArrayList<>(Arrays.asList(565,670)))	ArrayList that contain blacksmith's width
+ ArrayList<Integer> BLACK_SMITH_HEIGHT (= new ArrayList<>(Arrays.asList(343,577)))	ArrayList that contain blacksmith's height
+ ArrayList<Integer> BLACK_SMITH_HITBOX_X_OFFSET (= new ArrayList<>(Arrays.asList(155,110)))	ArrayList that contain blacksmith's hitbox offset in x-axis
+ ArrayList<Integer> BLACK_SMITH_HITBOX_Y_OFFSET (= new ArrayList<>(Arrays.asList(185,410)))	ArrayList that contain blacksmith's hitbox offset in y-axis
+ ArrayList<Integer> BLACK_SMITH_HITBOX_WIDTH (= new ArrayList<>(Arrays.asList(260,360)))	ArrayList that contain blacksmith's hitbox width
+ ArrayList<Integer> BLACK_SMITH_HITBOX_HEIGHT (= new ArrayList<>(Arrays.asList(160,170)))	ArrayList that contain blacksmith's hitbox height
+ ArrayList<Integer> BLACK_SMITH_FRAMES (= new ArrayList<>(Arrays.asList(4,6)))	ArrayList that contain blacksmith's frames
+ ArrayList<Integer>	ArrayList that contain blacksmith's

BLACK_SMITH_FRAMES_PER_SECOND (= new ArrayList<>(Arrays.asList(15,3))	frames per second
+ ArrayList<Integer> BLACK_SMITH_PAUSE_TIME (= new ArrayList<>(Arrays.asList(3,1)))	ArrayList that contain blacksmith's pause time

5.13.8. Class PerkStatus

- This class is used as a container of perk stats

Fields

Name	Description
+ float PRICE_UPGRADE_PERCENT (= 1.5f)	The multiplier for creating the upgradePrice
+ int PRICE_VARIATION (= 25)	The variation of the price
+ int COMMON_PRICE (= 150)	Price for Common Perk
+ int UNCOMMON_PRICE (= 200)	Price for Uncommon Perk
+ int RARE_PRICE (= 250)	Price for Rare Perk
+ float BERSERK_1_CRITERIA (= 0.3f)	The criteria of TIER1 BerserkPerk
+ float BERSERK_2_1_CRITERIA (= 0.4f)	The criteria of TIER2_1 BerserkPerk
+ float BERSERK_2_2_CRITERIA (= 0.2f)	The criteria of TIER2_2 BerserkPerk
+ int BERSERK_1_ADDATK (= 20)	The additional atk from TIER1 BerserkPerk
+ Int BERSERK_2_1_ADDATK (= 25)	The additional atk from TIER2_1 BerserkPerk
+ int BERSERK_2_2_ADDATK (= 40)	The additional atk from TIER2_2 BerserkPerk
+ float DISCOUNT_NORMAL_FAILCHANCE (= 0f)	The Normal Chance of failing from DiscountMasterPerk
+ float	The TIER2_2 Chance of failing from

	DISCOUNT_2_2_FAILCHANCE (= 0.4f)	DiscountMasterPerk
+	int DISCOUNT_1_DISCOUNT (= 15)	The discount from TIER1 DiscountMasterPerk
+	int DISCOUNT_2_1_DISCOUNT (= 25)	The discount from TIER2_1 DiscountMasterPerk
+	int DISCOUNT_2_2_DISCOUNT (= 50)	The discount from TIER2_2 DiscountMasterPerk
+	float EXTROVERT_TILES_X (= 4f)	The amount of tiles to check in X-Axis for ExtrovertPerk
+	float EXTROVERT_TILES_Y (= 0.25f)	The amount of tiles to check in Y-Axis for ExtrovertPerk
+	int EXTROVERT_CRITERIA (= 2)	The criteria of ExtrovertPerk
+	int EXTROVERT_1_ADDATKPERMONSTER (= 3)	The additional atk from TIER1 ExtrovertPerk
+	int EXTROVERT_2_1_ADDATKPERMONSTER (= 5)	The additional atk from TIER2_1 ExtrovertPerk
+	int EXTROVERT_2_2_ADDATKPERMONSTER (= 10)	The additional atk from TIER2_2 ExtrovertPerk
+	int EXTROVERT_2_2_MINUSATKEXTRA (= 12)	The constant additional atk from TIER2_2 ExtrovertPerk
+	int FATAL_1_ADDCRITDAMAGE (= 20)	The additional critDamage from TIER1 FatalAttackPerk
+	int FATAL_2_1_ADDCRITDAMAGE (= 35)	The additional critDamage from TIER2_1 FatalAttackPerk
+	int FATAL_2_2_ADDCRITDAMAGE (= 50)	The additional critDamage from TIER2_2 FatalAttackPerk
+	int FATAL_2_2_MINUSCRITRATE (= 5)	The constant additional critRate from TIER2_2 FatalAttackPerk

+ int FORTIFY_1_ADDDAMAGEDECREASE (= 15)	The additional damageDecrease from TIER1 FortifyPerk
+ int FORTIFY_2_1_ADDDAMAGEDECREASE (= 25)	The additional damageDecrease from TIER2_1 FortifyPerk
+ int FORTIFY_2_2_ADDDAMAGEDECREASE (= 10)	The additional damageDecrease from TIER2_2 FortifyPerk
+ int FORTIFY2_2_ADDATKDELAY (= 30)	The additional delay to all monsters from TIER2_2 FortifyPerk
+ float INTROVERT_TILES_X (= 6f)	The amount of tiles to check in X-Axis for IntrovertPerk
+ float INTROVERT_TILES_Y (= 0.25f)	The amount of tiles to check in Y-Axis for IntrovertPerk
+ int INTROVERT_1_BASEADDATK (= 12)	The base additional atk from TIER1 IntrovertPerk
+ int INTROVERT_1_MINUSADDATKPERMONSTER (= 4)	The decrement of additional atk from TIER1 IntrovertPerk
+ int INTROVERT_2_1_BASEADDATK (= 18)	The base additional atk from TIER2_1 IntrovertPerk
+ int INTROVERT_2_1_MINUSADDATKPERMONSTER (= 3)	The decrement of additional atk from TIER2_1 IntrovertPerk
+ int INTROVERT_2_2_BASEADDATK (= 30)	The base additional atk from TIER2_2 IntrovertPerk
+ int INTROVERT_2_2_MINUSADDATKPERMONSTER (= 15)	The decrement of additional atk from TIER2_2 IntrovertPerk
+ int JUKE_1_ADDEVADERATE (= 15)	The additional evadeRate from TIER1 JukeMasterPerk
+ int JUKE_2_1_ADDEVADERATE (= 25)	The additional evadeRate from TIER2_1 JukeMasterPerk

+ int JUKE_2_2_ADDEVADERATE (= 50)	The additional evadeRate from TIER2_2 JukeMasterPerk
+ int JUKE_2_2_MINUSDAMAGEDECREASE (= 40)	The constant additional damageDecrease from TIER2_2 JukeMasterPerk
+ float LUCKYMAN_NORMAL_TIERCHANCE (= 0.08f)	The normal chance of getting TIER2_1 or TIER2_2 perk from LuckyManPerk
+ float LUCKYMAN_2_1_TIERCHANCE (= 0.12f)	The TIER2_1 chance of getting TIER2_1 or TIER2_2 perk from LuckyManPerk
+ float LUCKYMAN_2_2_DROPCHANCE (= 0.1f)	The TIER2_2 increasing chance of monster to drop hp when died
+ float PERFECTION_1_CRITERIA (= 0.8f)	The criteria of TIER1 PerfectionistPerk
+ float PERFECTION_2_1_CRITERIA (= 0.7f)	The criteria of TIER2_1 PerfectionistPerk
+ float PERFECTION_2_2_CRITERIA (= 1f)	The criteria of TIER2_2 PerfectionistPerk
+ int PERFECTION_1_ADDATK (= 10)	The additional atk from TIER1 PerfectionistPerk
+ int PERFECTION_2_1_ADDATK (= 15)	The additional atk from TIER2_1 PerfectionistPerk
+ int PERFECTION_2_2_ADDATK (= 25)	The additional atk from TIER2_2 PerfectionistPerk
+ int PRECISION_1_ADDCRITRATE (= 10)	The additional critRate from TIER1 PrecisionStrikePerk
+ int PRECISION_2_1_ADDCRITRATE (= 20)	The additional critRate from TIER2_1 PrecisionStrikePerk
+ int PRECISION_2_2_ADDCRITRATE (= 30)	The additional critRate from TIER2_2 PrecisionStrikePerk

+ int PRECISION_2_2_MINUSCRITDAMAGE (= 20)	The constant additional critDamage from TIER2_2 PrecisionStrikePerk
+ int RAPID_NORMAL_ADDATKSPEED (= 15)	The normal additional attack speed for TIER1 RapidFirePerk
+ int RAPID_2_1_ADDATKSPEED (= 25)	The TIER2_1 additional attack speed for RapidFirePerk
+ int RAPID_2_2_ADDMOVEMENTSPEED (= 15)	The TIER2_2 additional movement speed for RapidFirePerk
+ int REINFORCED_1_ADDDEF (= 25)	The additional def for TIER1 ReinforcedPerk
+ int REINFORCED_2_1_ADDDEF (= 35)	The additional def for TIER2_1 ReinforcedPerk
+ int REINFORCED_2_2_ADDDEF (= 50)	The additional def for TIER2_2 ReinforcedPerk
+ int REINFORCED_2_2_MINUSMOVEMENTSPEED (= 25)	The constant additional movement speed for TIER2_2 ReinforcedPerk
+ float THORN_NORMAL_REFLECTCHANCE (= 0.6f)	The normal chance of ThornPerk to reflect damage
+ float THORN_2_1_REFLECTCHANCE (= 1f)	The TIER2_1 chance of ThornPerk to reflect damage
+ int THORN_2_2_SLOWTIME (= 3)	The TIER2_2 duration of ThornPerk for slowing a monster
+ int THORN_2_2_SLOW (= 80)	The TIER2_2 slow effect of ThornPerk
+ float TREASURE_NORMAL_REWARDCHANCE (= 0.3f)	The normal chance of getting double reward by TreasureHunterPerk
+ float TREASURE_2_1_REWARDCHANCE (= 0.5f)	The TIER2_1 chance of getting double reward by TreasureHunterPerk
+ float TREASURE_2_2_REWARDMULTIPLIER (= 1.25f)	The TIER2_2 multiplier for the coin amount in TreasureHunterPerk

+ int UNDEAD_NORMAL_ACTIVATION (= 3)	The normal duration that UndeadPerk activates
+ int UNDEAD_2_1_ACTIVATION (= 5)	The TIER2_1 duration that UndeadPerk activates
+ int UNDEAD_NORMAL_COOLDOWN (= 60)	The normal cooldown that UndeadPerk takes
+ int UNDEAD_2_2_COOLDOWN (= 30)	The TIER2_2 cooldown that UndeadPerk takes
+ int UNDEAD_2_1_ADDATK (= 20)	The additional atk of TIER2_1 UndeadPerk
+ int UNDEAD_2_1_ADDATKSPEED (= 50)	The additional attack speed of TIER2_1 UndeadPerk
+ VAMP_NORMAL_DRAINCHANCE (= 0.15f)	The normal chance of restoring hp by VampirismPerk
+ VAMP_2_2_DRAINCHANCE (= 0.25f)	The TIER2_2 chance of restoring hp by VampirismPerk
+ VAMP_NORMAL_DRAINPERHIT (= 0.01f)	The normal multiplier of restoring hp by VampirismPerk
+ VAMP_2_1_DRAINPERHIT (= 0.02f)	The TIER2_1 multiplier of restoring hp by VampirismPerk

5.13.9. Class PlayerStatus

- This class is used as a container of player stats

Fields

Name	Description
+ int START_COINS (= 0)	Amount of starter coin
+ int MAX_PERK_OBTAIN (= 6)	Maximum perk that player can obtain
+ int LOW_HP (= 200)	Low health
+ int HIGH_HP (= 300)	High health
+ int LOW_ATK (= 20)	Low attack damage
+ int HIGH_ATK (= 28)	High attack damage

+ int LOW_DEF (= 6)	Low defense
+ int HIGH_DEF (= 10)	High defense
+ float LOW_MOVEMENT_SPEED (= 3.5f)	Low movement speed
+ float MEDIUM_MOVEMENT_SPEED (= 4.5f)	Medium movement speed
+ float HIGH_MOVEMENT_SPEED (= 5.5f)	High movement speed
+ float DAGGER_NORMAL_ATTACK_SPEED_MULTIPLIER (= 1)	Speed multiplier for dagger's normal attack
+ float DAGGER_SPECIAL_ATTACK_SPEED_MULTIPLIER (= 1.5f)	Speed multiplier for dagger's special attack
+ float BOW_NORMAL_ATTACK_SPEED_MULTIPLIER (= 2.3f)	Speed multiplier for bow's normal attack
+ float BOW_SPECIAL_ATTACK_SPEED_MULTIPLIER (= 1)	Speed multiplier for bow's special attack
+ float BROADSWORD_NORMAL_ATTACK_SPEED_MULTIPLIER (= 1.7f)	Speed multiplier for broadsword normal attack
+ float BROADSWORD_SPECIAL_ATTACK_SPEED_MULTIPLIER (= 2)	Speed multiplier for broadsword special attack
+ int LOW_EVADE_RATE (= 5)	Low evade rate
+ int HIGH_EVADE_RATE (= 15)	High evade rate
+ int LOW_CRIT_RATE (= 5)	Low critical rate
+ int HIGH_CRIT_RATE (= 15)	High critical rate
+ int LOW_CRIT_DAMAGE (= 50)	Low critical damage
+ int HIGH_CRIT_DAMAGE (= 75)	High critical damage

+ int NORMAL_MAX_DASH (= 2)	Default max dash
+ int LOW_MAX_DASH (= 1)	Low max dash
+ float DEFAULT_DASH_DISTANCE (= 5)	Default dash distance
+ int DEFAULT_DASH_TIME (= 20)	Default dash time
+ float DASH_COOLDOWN (= 2)	Defaut dash cooldown
+ float TIME_CAN_SECOND_ATTACK (= 0.375f)	Time duration that player can still attack with a second attack after first attack
+ float PLAYER_DAMAGE_COOLDOWN (= 0.125f)	Time duration that player can't receive any damage after being damaged

5.13.10. Class SoundLoader

- This class is used for loading sound in the game

Fields

Name	Description
- String name	Name for the SoundLoader's object
- MediaPlayer mediaPlayer	MediaPlayer of this object
- AudioClip audioClip	AudioClip of this object
- ArrayList<String> BACKGROUNDS (= new ArrayList<>(Arrays.asList("StartSceneSong", "ClassSelectionSceneSong", "FightSceneSong1", "FightSceneSong2", "FightSceneSong3", "FightSceneSong4", "FightSceneSong5", "FightSceneSong6", "MarketSceneSong", "BossSceneSong")))	ArrayList containing the background song's name
- boolean isPlay (= false)	Boolean to check whether the sound plays or not

Constructors

Name	Description
+ SoundLoader(String name, String url)	Constructor for SoundLoader - Set name, mediaPlayer and audioClip

Methods

Name	Description
+ void play()	Play the sound
+ void stop()	Stop the sound
+ String getName()	Getter for name
+ MediaPlayer getMediaPlayer()	Getter for mediaPlayer